1989

Extension of aggregation and shrinkage techniques used in the estimation of Marine Corps Officer attrition rates.

Misiewicz, John M.

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

EXTENSION OF AGGREGATION AND SHRINKAGE TECHNIQUES USED IN THE ESTIMATION OF MARINE CORPS OFFICER ATTRITION RATES

by

John M. Misiewicz

September 1989

Thesis Advisor: Robert R. Read

Approved for public release; distribution is unlimited

Unclassified

security classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification Unclassified

1b Restrictive Markings

2a Security Classification Authority

2b Declassification Downgrading Schedule

3 Distribution Availability of Report
Approved for public release; distribution is unlimited.

4 Performing Organization Report Number(s)

5 Monitoring Organization Report Number(s)

6a Name of Performing Organization
Naval Postgraduate School

6b Office Symbol *(if applicable)* 30

7a Name of Monitoring Organization
Naval Postgraduate School

6c Address *(city, state, and ZIP code)*
Monterey, CA 93943-5000

7b Address *(city, state, and ZIP code)*
Monterey, CA 93943-5000

8a Name of Funding Sponsoring Organization

8b Office Symbol *(if applicable)*

9 Procurement Instrument Identification Number

8c Address *(city, state, and ZIP code)*

10 Source of Funding Numbers

Program Element No	Project No	Task No	Work Unit Accession No

11 Title *(include security classification)* EXTENSION OF AGGREGATION AND SHRINKAGE TECHNIQUES USED IN THE ESTIMATION OF MARINE CORPS OFFICER ATTRITION RATES

12 Personal Author(s) John M. Misiewicz

13a Type of Report
Master's Thesis

13b Time Covered
From To

14 Date of Report *(year, month, day)*
September 1989

15 Page Count
114

16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes

Field	Group	Subgroup

18 Subject Terms *(continue on reverse if necessary and identify by block number)*
aggregation, attrition rate estimation, empirical Bayes

19 Abstract *(continue on reverse if necessary and identify by block number)*

In this thesis we treat the "small cell" problem encountered when building an attrition rate generator for large-scale manpower flow models, specifically for the USMC Officer Corps. Such models have a large number of low-inventory (i.e. small) personnel cells. This presents a dilemma: on one hand we want to preserve as much fidelity as possible in our work by preserving a great deal of detail in each cell; on the other hand our statistical estimation techniques require larger cell sample sizes than intrinsically occur cell-by-cell in actual sample data. Our approach to producing stable attrition rates for such cells involves two efforts: (i) the aggregation of cells into groups that exhibit homogeneity of attrition behavior, and (ii) the development of "shrinkage" estimation techniques for use in the individual groups. A heuristic algorithm is developed and tested to treat the aggregation problem. Empirical Bayes methods are developed to serve the multi-cell estimation requirements needed to preserve the fidelity. Cross validation techniques are used to verify these methods.

The present work builds upon the results of previous studies; we integrate what was learned into a coherent package that is ready for use.

20 Distribution Availability of Abstract
☒ unclassified unlimited ☐ same as report ☐ DTIC users

21 Abstract Security Classification
Unclassified

22a Name of Responsible Individual
Robert R. Read

22b Telephone *(include Area code)*
(408) 646-2382

22c Office Symbol
55Re

DD FORM 1473,84 MAR

83 APR edition may be used until exhausted
All other editions are obsolete

security classification of this page

Unclassified

Approved for public release; distribution is unlimited.

Extension of Aggregation and Shrinkage Techniques Used in the Estimation of Marine Corps Officer Attrition Rates

by

John M. Misiewicz
Major, United States Marine Corps
B.S., Pennsylvania State University, 1976
M.S., University of Southern California, 1983

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1989

ABSTRACT

In this thesis we treat the "small cell" problem encountered when building an attrition rate generator for large-scale manpower flow models, specifically for the USMC Officer Corps. Such models have a large number of low-inventory (i.e. small) personnel cells. This presents a dilemma: on one hand we want to preserve as much fidelity as possible in our work by preserving a great deal of detail in each cell; on the other hand our statistical estimation techniques require larger cell sample sizes than intrinsically occur cell-by-cell in actual sample data. Our approach to producing stable attrition rates for such cells involves two efforts: (i) the aggregation of cells into groups that exhibit homogeneity of attrition behavior, and (ii) the development of "shrinkage" estimation techniques for use in the individual groups. A heuristic algorithm is developed and tested to treat the aggregation problem. Empirical Bayes methods are developed to serve the multi-cell estimation requirements needed to preserve the fidelity. Cross validation techniques are used to verify these methods.

The present work builds upon the results of previous studies; we integrate what was learned into a coherent package that is ready for use.

THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

I. INTRODUCTION 1
A. GENERAL 1
B. BACKGROUND 2
C. ORGANIZATION 3
D. SMALL CELL PROBLEM 4
E. DATA BASE 5

II. CELL AGGREGATION 7
A. GENERAL 7
B. BACKGROUND 7
C. EXPANSION METHOD 9
D. AGGREGATION METHOD 14

III. ESTIMATION METHODS 16
A. GENERAL 16
B. EMPIRICAL BAYES 17
1. Transformed Scale 17
2. Original Scale 20
C. EFRON-MORRIS METHOD 23
D. VECTOR METHOD 25

IV. CROSS VALIDATION 28
A. GENERAL 28
B. MEASURES OF EFFECTIVENESS 28
1. Mean Absolute Deviation 28
2. Chi Square 29
3. Mean Squared Error 30
4. Vector Method MOEs 31
C. TEST CASES 31
D. RESULTS 35

V. CONCLUSIONS AND RECOMMENDATIONS 45
A. CONCLUSIONS 45
B. RECOMMENDATIONS 46

APPENDIX A. AGGREGATION ALGORITHMS 47
A. HEURISTIC ALGORITHM 47
B. INTEGER LINEAR PROGRAM 47

APPENDIX B. COMPUTER PROGRAMS 49
A. GENERAL 49
B. MAIN PROGRAM AND AGGREGATION SUBROUTINES 51
C. ESTIMATION SUBROUTINES 73
D. VECTOR METHOD SUBROUTINE 88
E. EXEC PROGRAM 94
F. SAMPLE DATA FILE 95

APPENDIX C. SAMPLE OUTPUT 98
A. GENERAL 98
B. SAMPLE OUTPUT (TEST CASES 1-30) 98
C. SAMPLE OUTPUT (VECTOR TEST CASES) 101

REFERENCES 102

INITIAL DISTRIBUTION LIST 104

LIST OF TABLES

Table 1. MOS GROUPS 11
Table 2. YCS EXPANSION BOUNDS 12
Table 3. TEST CASES FOR METHODS 1-5 33
Table 4. TEST CASES FOR VECTOR METHOD 34
Table 5. SUMMARY OF RESULTS (CASES 1-10) 40
Table 6. SUMMARY OF RESULTS (CASES 11-20) 41
Table 7. SUMMARY OF RESULTS (CASES 21-30) 42
Table 8. SUMMARY OF RESULTS (CASES 7 AND 9 EXPANDED) 43
Table 9. SUMMARY OF RESULTS (VECTOR METHOD) 44

I. INTRODUCTION

A. GENERAL

The Officer Planning and Utility System (OPUS), a comprehensive and fully integrated manpower management system, is currently being implemented by the U.S. Marine Corps (Decision System Associates, 1986). This system contains a set of computer-based manpower planning models and is used by the Officer Plans Section (MPP-30), Headquarters, U.S. Marine Corps, to produce several manpower planning documents. The system must be able to accurately predict personnel attrition, i.e., officers leaving the service for purposes such as resignation, retirement, discharge, disability, or release. The forecasting of attrition is accomplished by the Marine Corps Officer Rate Projector (MCORP), developed by the Navy Personnel Research and Development Center (NPRDC), San Diego, California (NPRDC, 1985).

The attrition rate generator developed by NPRDC calculates empirical attrition rates using historical data with user-defined weights and threshold parameters (Siegel, 1983). This subjective input makes the current generator susceptible to unintentional misuse.

In support of MCORP, Professor Robert R. Read of the Naval Postgraduate School has been working on the "small cell" problem: applying multiparameter statistical estimation schemes to estimating attrition when there is low personnel inventory, or small cells, which generally exhibit unstable empirical rates.

A comment on terminology is in order. By attrition rate generator we mean methodology for estimating attrition probabilities for the various cells. The expression "empirical rates" refers to the ratio of leavers to inventory for each cell, unmodified by any information contained in "neighboring" cells. In contrast to this, the expression "empirical Bayes" refers to Bayes estimators whose prior parameters are estimated from data.

Accurate forecasting of losses is extremely important to the manpower planner. Overestimating losses causes excess accessions, promotion delays, underutilized personnel and increased costs, while underestimation causes personnel shortages and decreased readiness. The problem is compounded in that all but a few accessions must start at the bottom, i.e., Second Lieutenant, and work their way up to the higher ranks only after many years of service. For example, if a shortage of Lieutenant Colonels arises, it can

only be remedied by promoting more Majors, which has a rippling effect down the rank structure.

B. BACKGROUND

There have been seven Master's theses over the past four years which have studied various aspects of the attrition estimation problem. A concise summary of these works is given by Read (NPS Report NPS55-88-006, 1988, pp.16-23). These studies can be grouped into three general areas: shrinkage methods, cell aggregation and peripheral studies.

The application of a shrinkage method begins by identifying a number of personnel inventory cells, followed by the development of the empirical rates for individual cells and a weighted grand mean of these empirical rates. The final estimate for a cell is a convex combination of its empirical rate and the grand mean. There are numerous methods for accomplishing this, several of which have been applied in previous studies.

Tucker was the first to investigate the application of these methods to attrition estimation. He compared traditional estimators to the James-Stein estimator and the minimax estimator for a few selected grades and occupational fields. His results gave strong support to the James-Stein estimator; minimax was discarded as being too conservative for small cell use. However, there remained pockets of instability for which goodness-of-fit tests failed. (Tucker, 1985)

Following Tucker was Robinson, who introduced the Efron-Morris limited translation shrinkage alternative to augment the James-Stein estimator. These methods were evaluated with a broader set of test cases. Robinson was able to confirm Tucker's results, but the limited translation option failed to provide any consistent relief in the unstable areas. (Robinson, 1986)

The final application of shrinkage methods to estimating officer attrition rates was undertaken by Dickinson. He applied the previously used methods and an empirical Bayes estimator to a new and refined data base. Improved results were obtained, but the instability remained. Dickinson also performed some exploratory side studies dealing with the Freman-Tukey transform and the use of empirical Bayes methods that allow non-uniform shrinkage, both of which provided the impetus for the present study. (Dickinson, 1988)

These three studies used ad hoc methods to deal with the second general area of study--cell aggregation. Aggregation of cells with low personnel inventory into sets of cells, often of larger inventory, is required when applying these shrinkage methods. The

desire is to use cells which exhibit similar attrition behavior. Two previous studies have investigated this area.

Amin Elseramegy used the Classification and Regression Trees (CART) program, which at the time was newly acquired by the Naval Postgraduate School, in an attempt to form aggregates of cells that exhibited homogeneous attrition behavior. Several difficulties in using this program were encountered, e.g., because of insufficient memory allocation he found it necessary to partition the data base into nine sets and apply CART to each. The resulting aggregations were generally unusable. (Amin Elseramegy, 1985)

Major breakthroughs in cell aggregation were made by Larsen. He applied a hierarchical clustering algorithm to the new data base. The resulting rules for building aggregates are well defined and especially viable from an intuitive point of view. Larsen's work provides the framework for the cell aggregation method developed in Chapter II of this thesis. (Larsen, 1987)

The remaining two theses of the seven were peripheral studies which applied alternate methods to attrition estimation. Hogan attempted multi-year forecasting using exponential smoothing; the smoothing constants were rather unusual and extreme and his results inconsistent (Hogan, 1986). Yacin applied logistic regression in the attempt to develop an attrition rate scheme; the only new results were the identification of some areas that exhibited similar attrition behavior (Yacin, 1987).

This thesis is the first to integrate the two main areas of study. Whereas previous studies of shrinkage methods have used ad hoc aggregation schemes, we now combine the implementation of a defensible aggregation method with empirical Bayes estimators. Moreover, these are applied to a larger and more refined data base. The results have been quite promising in that we have achieved greater stability in attrition rate estimation; we have defined guidelines for a heuristically appealing aggregation scheme; and we have acquired an increased understanding of the data base and developed more efficient ways to use it.

C. ORGANIZATION

The remainder of this introductory chapter provides a more detailed description of the small cell problem and the data base. The aggregation problem is discussed and the proposed aggregation method is presented in Chapter II.

The shrinkage estimation methods, generally classified as empirical Bayes type estimators, are described in Chapter III. Several variations are presented to allow

comparison and to gain further insight into their performance. Testing of these methods is important but for practical purposes must be carried out using sampling methods. The rationale used to select test cases, the cross validation techniques, and the measures of effectiveness used to evaluate the results are discussed in Chapter IV. A discussion of the results of the cross validation is also included in this chapter.

Finally, conclusions and recommendations based on these results are contained in Chapter V.

D. SMALL CELL PROBLEM

Marine Corps officers can be classified and thus partitioned by several attributes. The major partitioning of officers is by grade, years commissioned service (YCS), and military occupational specialty (MOS). Grade describes the position an officer holds in the service. The numbers of officers in each higher grade have a pyramid structure, i.e., there are more officers in the lower grades than in the higher grades. YCS is the total number of years served since becoming a commissioned officer. There is a strong correlation between grade and YCS since an officer generally moves up the grade structure as he gains in YCS. MOS is a four-digit code identifying the specific skill for which a Marine is trained. MOS need not remain constant over an officer's career, although most changes in MOS occur in the early years of commissioned service. An officer has a single primary MOS, however as he develops new job skills he may be assigned one or more additional MOSs.

For many purposes, partitioning by grade, YCS and MOS is sufficient. However in some applications additional refinement by service component, commissioning source, sex, race or education level may be necessary. Service component consists of three categories: regular officers, reserve officers, and reserves who have augmented to become regulars. It is strongly correlated with commissioning source, i.e., an officer receives a regular or reserve commission depending upon the commissioning source. Both affect an officer's initial service obligation, which is generally three to five years (except aviators, whose obligation is dependent upon the amount of flight training). Officers who receive a reserve commission normally serve three to four years active duty (except aviators), by the end of which they must have either augmented into the regular force or are then separated from active service.

These cross-classifications may be viewed as breaking the officer population into a multidimensional array, with each specific intersection of the classifications called a cell. The total number of possible cells is quite large, on the order of 10^6. Many of these cells

are structurally infeasible in that no officer could possibly fit the cell characteristics, e.g., there are no Majors with two years commissioned service. The total officer inventory of approximately 20,000 officers is partitioned by the remaining feasible cells; some cells have inventory as large as 50, however most have less than five. An officer's characteristics are dynamic, i.e., as an officer moves through the grade, YCS and MOS structure he moves from one cell to another. As a result, the inventory of the feasible cells is also dynamic and fluctuates between zero and low inventory (less than five) over time.

These sparsely populated cells have very unstable empirical attrition rates. For example, a cell whose inventory is two officers of which one leaves the service during a given time period yields a 50% empirical attrition rate, whereas a cell whose inventory is one officer who remains in service during the same time period yields a 0% attrition rate. It is obvious that neither of these empirical estimations provides a usable attrition rate. Furthermore, these two rates could change dramatically during the next time period, typifying their instability.

Even when more modern estimation techniques (e.g. shrinkage) are applied, these small cells can still create statistical instability, thereby producing intolerably variable attrition rate estimates. The problem then is how to deal with these low inventory cells, or "small cells" in order to achieve stability.

E. DATA BASE

In this thesis we benefit from a refined data base compiled by NPRDC and made available to the Naval Postgraduate School in 1987. This data base, used by Larsen in his aggregation work (Larsen, 1987), was not available for the previous estimation studies at NPS.

The new data base provides more detailed information about the officer population. The grade structure now allows separation of Limited Duty Officers (LDO) as well as Warrant Officers (WO) from unrestricted officers. Officers who have failed selection to the next higher grade can also be identified. YCS is listed instead of length of service (LOS), which became ambiguous when dealing with officers who have prior enlisted service. MOS can now be broken out completely into 236 MOSs or summarized by the 39 occupational fields. Service component and commissioning source are both new categories. Other new categories that are not considered here are education level, race, additional MOSs, and military schools completed. Larsen gives a complete description of the classifications (Larsen, 1987, pp.66-82).

The data base also allows attrition to be broken out by retirement, release, discharge, resignation, etc., but for our immediate purpose we are only concerned with the total number of losses for any reason.

This refinement of the data presents a dichotomy: we can now break the data into more definitive cells to search for homogeneous attrition behavior and stability in estimation, but this leads to an even greater number of low inventory cells.

The new data base contains ten years of inventory and attrition data from the period 1977-1986, a significant improvement from the previous seven year data base covering the period 1977-1983. The inventory data is now obtained from quarterly vice yearly snap-shots of the officer population. The attrition data is annualized, i.e., the attrition count for a cell reflects the number of personnel who leave the service at any time during the year. Attritions are credited to the cell which the officer occupies at the time he leaves.

Two problems arise from this quarterly versus annualized data. First, it is possible for a cell to record zero inventory via the snap-shots, yet be credited with one or more attritions. To avoid this situation, the cell inventory used in all calculations is defined to be the larger of the inventory and the attrition count. This ensures that the inventory for a cell is as least as large as its recorded number of leavers. (This override occurs infrequently; a more sophisticated treatment would require significant model enhancement.) Second, to use the inventory and attrition data together we must divide the inventory data by four. This poses a philosophical problem when invoking a binomial model: the sample size may not be integral. However, for our application the usual mean and variance formulas are usable and can still serve in the interpolative sense.

II. CELL AGGREGATION

A. GENERAL

The aggregation problem takes on new meaning with the use of shrinkage estimators. Originally, aggregation had only one concern: how to pool cells together into a **single** cell in order to meet a user-defined minimum inventory threshold. This single aggregated cell was then used to determine the attrition rate estimate for the original, unaggregated cell. In this way an estimated value for a cell is obtained by using the grand mean for many cells.

The empirical Bayes multiparameter estimation techniques provide a way to compromise, using both the stability of a grand mean and the specific information of an individual cell. Now we pool cells together and obtain a **number** of cells that meet the user-defined minimum inventory threshold. It is important to note that we should be able to use a lower inventory threshold with empirical Bayes, thus retaining individual cell behavior to a greater extent. It is also important to use cells with homogeneous attrition behavior in the aggregation process.

B. BACKGROUND

The aggregation method currently used by MCORP is called the Small Cell Override Methodology (NPRDC, 1985, Appendix H). It is used to solve the original aggregation problem, i.e., if a cell is below the user-defined threshold, then cells are adjoined to the original cell until the threshold is met. The process for selecting cells for adjunction is rather crude, and large-scale with only a few levels (prior to using the entire officer corps). The attrition rate estimate for the original cell is the empirical rate from this aggregated cell.

To begin the process, the user defines a cell for which an attrition rate estimate is required by grade, YCS and MOS. The user also defines the minimum cell inventory threshold (and other parameters which are not relevant here). If the cell he identifies meets the threshold, no aggregation is required and the empirical attrition rate is determined. If the cell is below the threshold, additional cells must be added until the threshold is met.

This search for additional cells occurs by expanding by YCS and MOS, with grade remaining fixed throughout. Expanding in this sense means changing the YCS or MOS parameter to identify the additional cells to be added to the original cell. Initially, the

single cell is expanded by YCS. For example, if the original cell's grade/YCS/MOS was Capt/7/0802, the cells identified by Capt/6/0802, Capt/8/0802, etc., are added sequentially until the threshold is met. This YCS expansion has an upper bound at the 20 YCS point; an obvious boundary for attrition behavior due to retirement eligibility. If the original cell's YCS is above 20, then 20 would serve as the lower YCS bound.

If the threshold is not met after the YCS expansion, the override method starts over with the original cell and expands by MOS. Each MOS belongs to one of nine MOS groups which are defined along traditional Marine Corps functional areas, e.g., all helicopter pilot MOSs are grouped together as are all combat support MOSs. MOS expansion adds those cells identified by the MOSs in the same MOS group as the MOS of the original cell for the original YCS and grade. If the threshold is not met, all the MOSs in the MOS group are expanded by YCS in the same manner as the YCS expansion discussed previously.

If this MOS group and YCS expansion is unsuccessful, the override method starts over with the original cell and expands by all MOSs for the original grade and YCS. If necessary, all the MOSs are expanded by YCS as before.

Cell aggregation using this expansion method can potentially include all MOSs and YCS bounded only at the 20 year point. The desire to aggregate using cells with homogeneous attrition behavior is obviously compromised. Larsen provides a more comprehensive description of the current method (Larsen, 1987, pp.16-22).

Larsen examined attrition behavior in the MOS and YCS structure. He applied a hierarchical clustering algorithm in an attempt to find MOSs and YCSs that displayed homogeneous attrition behavior. He confirmed the belief that YCS is an important factor. The YCS expansion bounds he proposed reflect points at which officers reach the end of their initial service obligation as well as when they are eligible for retirement, which makes them especially viable from an intuitive point of view. Larsen also found that some MOSs did not cluster strictly by functional areas. This was especially significant in the aviation community. Whereas the previous data base allowed aviators to be considered only as one occupational field, the refined MOS information was able to identify six distinct homogeneous groups of aviators.

Larsen uses these results to define more refined MOS groups and YCS boundaries. To avoid the giant expansion leap from MOS group to all MOSs, he proposed a hierarchy of small MOS groups, large MOS groups and major MOS groups developed by observing which MOSs tend to exhibit similar attrition behavior. Homogeneity is greatest within the MOS group, and becomes successively worse as we move to the large

MOS group and then the major MOS group. Each MOS is assigned to a small MOS group. Small MOS groups combine to make a large MOS group, and large MOS groups combine to make a major MOS group.

Each small MOS group is assigned a set of YCS expansion bounds. Due to the different attrition behavior of the small MOS groups with respect to YCS, three different sets of YCS expansion bounds are proposed.

Initial expansion is by YCS within the specified boundaries, with grade and MOS held constant. If more expansion is required, we retain this aggregated cell and expand by small MOS group for the original grade and YCS. If the aggregated cell is still below the threshold, the MOSs in the small MOS group are expanded by YCS. Subsequent expansion to large MOS group and YCS, and major MOS group and YCS is accomplished until the threshold cell inventory is met.

Unlike the current expansion method, expansion using Larsen's proposed method will not cross defined MOS groups or YCS bounds to ever include all MOSs and YCSs bounded only at the 20 year point. Larsen provides a more detailed description of his recommended expansion rules (Larsen, 1987, pp.45-61).

C. EXPANSION METHOD

We now address the methods used to obtain the cells required for use with empirical Bayes estimation techniques. Expansion continues to mean finding more cells to be used, however we no longer simply add these cells to the original cell to form a single aggregated cell. The cells identified by the expansion process are now aggregated together to produce a number of cells. After the discussion of the expansion process in this section, an actual aggregation scheme is introduced in the next section.

To begin the estimation process, the manpower planner defines a specific cell by grade, YCS and MOS. The attributes service component and commissioning source are also included as possible cell descriptors for the purposes of this study. All other descriptors listed in the data base--sex, education level, additional MOSs, race and military schools--are ignored. Loss types are considered as a combined total, i.e., in this study we do not discriminate among the various types of losses. The first three user-defined descriptors--grade, YCS and MOS--are single-value inputs. The last two descriptors, service component and commissioning source, can be single values, or either one of them can be treated as a vector of values for each single cell. This vector is collapsed (total the components) during the aggregation process, i.e., all records which meet any of the vector's values are included in the same cell. As in the previously described expansion

methods, only YCS and MOS change during expansion, the remaining cell descriptors remain constant.

To use shrinkage techniques, the amount of expansion required not only depends upon the minimum cell inventory threshold but also upon a new input parameter: the threshold number of cells. These two parameters are denoted:

1. T_0 - cell inventory threshold. The minimum average inventory for a cell obtained by averaging the cell inventory over the ten years of data.
2. K_0 - threshold number of cells. The minimum number of aggregated cells whose inventory exceeds T_0. These aggregated cells are the input cells for the empirical Bayes techniques.

For example, if $T_0 = 5.0$ and $K_0 = 10$, the expansion algorithm continues until at least ten aggregated cells, each with average inventory 5.0 or larger are obtained. Since we are concerned primarily with the small cell problem the values of T_0 and K_0 used are selected to range from five to 30. It is also presumed that T_0 is less than or equal to K_0. These threshold values can certainly exceed 30 for other applications, however the resulting cells are not considered small and their attrition behavior most likely would not be as unstable, therefore not requiring special attention.

Prior to explaining the expansion process, we first define the MOS groups and YCS bounds. We have adopted much of Larsen's work in this area; many of the changes are minor but are necessary for implementation purposes.

The general idea of a hierarchy of MOS groups is repeated, as shown in Table 1. Each MOS belongs to a small MOS group, a large MOS group and a major MOS group. Listed are 14 small MOS groups, which combine to make six large MOS groups, which combine to make four major MOS groups. For example, small MOS groups one and two form large MOS group one, and small MOS groups three through six form large MOS group two. Large MOS groups one and two, which collectively contain small MOS groups one through six, make up major MOS group one. Major MOS group one contains only ground MOSs, and major MOS group two contains only aviation MOSs. Major MOS groups three and four are special cases as discussed below.

A subjective decision was made to keep the ground MOSs in groups defined along the more traditional functional areas. This is reflected in small MOS groups one through six. For estimation purposes it is advantageous if the cell inventories are not too variable in size (Carter and Rolph, 1974, p.882). It is also desirable to avoid having too many MOSs in each small MOS group. This allows the expansion to occur more gradually, and is especially important for small values of T_0 and K_0. As a result, MOS 0302

Table 1. MOS GROUPS

Group Name	MOSs	Small MOS Group	Large MOS Group	Major MOS Group
Combat	0302	1	1	1
Combat Support	0802 1302 1802 1803	2		
Combat Service 1	0180 0202 2502 2602	3	2	
Combat Service 2	3415 4002 4302 5803	4		
Combat Logistics	0402 3002 3060 3502 6002	5		
Air Control	7204 7208 7210 7320	6		
Fixed Wing Pilots	7501 7511 7522 7542 7543 7545 7576	7	3	2
F-18 Pilots	7521 7523	8		
Rotary Wing Pilots +	7556 7557 7562 7564 7565 7566 7587	9	4	
Naval Flight Officers +	7508 7509 7563 7581 7583 7584 7585 7586 7588	10		
Basic Ground	0101 0201 0301 0401 0801 1301 1801 2501 2601 3001 3401 3501 4001 4301 4401 5801 6001 7201 7301 9901	11	5	3
Student Aviators	7580 7597 7598 7599	12		
Basic Pilots	7500 7510 7520 7540 7550 7560 7575	13		
Lawyers	4402	14	6	4

(infantry) is placed alone in a small MOS group. This MOS contains approximately 15% of the total officer population, and therefore its respective cells normally contain large inventory. The MOSs in small MOS group two also contain fairly large inventory, therefore are grouped together and their first expansion is with MOS 0302. The remaining ground MOSs in small MOS groups three through six have similarly small inventory.

The aviation small MOS groups (seven through ten) remain relatively unchanged from Larsen's recommendations. MOS 7564 (CH-53 pilot), was removed from a ground MOS group and added to small MOS group nine, which reflects its functional area. MOSs 7551 (C-9 pilot), 7552 (TC-4C pilot), 7555 (UC-12B pilot) and 7559 (CT-39 pilot)

were deleted since they are not primary MOSs. MOS 7530 (basic pilot VMFA (F-4)) was deleted since it is not a current MOS. (MCO P1200.7G, 1988)

Officers who have not acquired sufficient schooling or field experience to qualify for a primary MOS listed in small MOS groups one through ten are gathered together as basic officers or students in small MOS groups 11-13. These officers are generally second lieutenants or junior first lieutenants with three or fewer YCS. They are disregarded for the remainder of the study because their attrition rates are extremely low; probably because none of the officers in these groups have reached the end of their initial obligations.

MOS 4402 (lawyers) is considered a special case and is not addressed in this study.

All MOSs listed in Table 1 are primary MOSs for unrestricted officers as listed in the current Military Occupational Specialties Manual (MCO P1200.7G, 1988). It would be a logical and relatively simple extension of this table to create additional groups containing LDO and WO MOSs. These grades are not considered in this study and therefore their respective MOSs are excluded from the table.

Several of these seemingly ad hoc decisions to alter Larsen's recommended MOS groups are due to the YCS expansion bounds shown in Table 2. Every effort was made to group MOSs with similar YCS expansion bounds to allow for feasible implementation of the expansion algorithm. This is especially applicable when expanding to large and major MOS groups.

Table 2. YCS EXPANSION BOUNDS

MOS Group	Small MOS Groups	Bounded YCS Groups
Fixed Wing Pilots, F-18 Pilots, Lawyers	7, 8, 14	(1-6, 8-19) (7) (20-25) (26)
Rotary Wing Pilots, Naval Flight Officers	9, 10	(1-5, 8-19) (6,7) (20-25) (26)
All Others	1-6, 11-13	(1-3, 6-19) (4,5) (20-25) (26)

The YCS expansion bounds reflect the maximum expansion allowed from the initial YCS defined by the user. For example, if the original cell's grade/YCS/MOS is Capt/9/7501, we see from Table 1 that this MOS belongs to small MOS group seven. Thus its YCS expansion bounds are listed on the first line of Table 2. The value of nine

for YCS falls in the first YCS range, thus we could expand using all YCSs from one through 19, excluding seven. If the YCS for this original cell had been seven, **no** YCS expansion would be allowed.

These YCS expansion bounds are used with the MOS groups to define the additional cells which can be used with the original cell to obtain the required number of cells, K_0, each with minimum average inventory, T_0. The expansion stages are:

1. Stage 1 - Locate the small MOS group which contains the user-defined MOS. The initial cells are those specified by the MOSs in this group for the user-defined YCS, grade, service component and commissioning source (grade, service component and commissioning source remain fixed throughout the expansion process and thus are not repeated). These cells are aggregated to obtain cells with average inventory greater than or equal to T_0. After aggregation, if the number of cells is greater than K_0, stop, otherwise go to Stage 2.
2. Stage 2 - Expand by incrementing YCS (YCS-1, YCS+1, YCS-2, YCS+2, etc.) within the YCS bounds listed in Table 2 for all MOSs in the small MOS group. After **each** YCS increment, aggregate the cells to obtain cells with average inventory greater than or equal to T_0. After aggregation, if the number of cells is greater than K_0, stop, otherwise continue to increment by YCS. If the YCS bounds are reached before obtaining enough aggregated cells, retain the cells identified in Stages 1 and 2 and go to Stage 3.
3. Stage 3 - Expand to the large MOS group for the **single** user-defined YCS. Aggregate the cells to obtain cells with average inventory greater than or equal to T_0. After aggregation, if the number of cells is greater than K_0, stop, otherwise go to Stage 4.
4. Stage 4 - Expand by incrementing YCS for the large MOS group. After **each** YCS increment, aggregate the cells to obtain cells with average inventory greater than or equal to T_0. After aggregation, if the number of cells is greater than K_0, stop, otherwise continue to increment by YCS. If the YCS bounds are reached before obtaining enough aggregated cells, retain the cells identified in Stages 1 through 4 and go to Stage 5.
5. Stage 5 - Expand to the major MOS group for the **single** user-defined YCS. Aggregate the cells to obtain cells with average inventory greater than or equal to T_0. After aggregation, if the number of cells is greater than K_0, stop, otherwise go to Stage 6.
6. Stage 6 - Expand by incrementing YCS for the major MOS group. After **each** YCS increment, aggregate the cells to obtain cells with average inventory greater than or equal to T_0. After aggregation, if the number of cells is greater than K_0, stop, otherwise continue to increment by YCS. If the YCS bounds are reached before obtaining enough aggregated cells, **stop**. No more expansion is allowed. Inform the user that the thresholds are unattainable. Do not cross any major MOS group or YCS bounds.

Two points about the expansion process are emphasized. First, we retain the cells identified by all previous stages as we progress to the next stage. As stated before, the degree of homogeneity decreases as we move from small to large to major MOS groups.

Thus we want to locate as many cells from the small MOS group as possible before we expand to the large MOS group, and then locate as many cells from the large MOS group as possible before expanding to the major MOS group. The YCS expansion for each group may be different, e.g., the small MOS group may be expanded by all YCSs within the given YCS range, but the large MOS group may only be expanded by a few YCSs before the thresholds are attained.

The second point is that, when aggregating cells, any aggregation that was performed previously is discarded and all cells currently identified are pooled and made available for aggregation. This affords the aggregation algorithm greater flexibility and could create more aggregated cells than if the aggregated cells from previous stages were left intact, thereby keeping the amount of expansion to a minimum.

The aviation small MOS groups (seven through ten) make up the only major MOS group (two) that contains different YCS bounds, i.e., small MOS groups seven and eight have different YCS expansion with regard to year six than do small MOS groups nine and ten. To implement the expansion algorithm in a computer program, this difference is overcome by using the YCS bounds for the original user-defined MOS. For example, suppose MOS 7501 from small MOS group seven is the original MOS. If MOS expansion continues into the major MOS group, the MOSs in small MOS groups nine and ten would follow small MOS group seven's YCS expansion bounds.

In summary, this method of grouping MOSs should provide greater homogeneity among cells which are used in estimating attrition rates. Unlike the current method, ground and aviation MOSs are never used together. The YCS bounds provide a logical and effective way to treat periods of different attrition behavior. However, the greater the expansion the less homogeneous the cells become, which should be kept foremost in mind when setting the threshold parameters.

D. AGGREGATION METHOD

While the expansion steps are being undertaken in order to achieve the threshold levels specified by the user, those cells with inventory less than T_0 must be gathered up into larger, aggregated cells whose combined inventory exceeds T_0. In order to limit the expansion to as few additional MOSs and YCSs as possible, we desire to maximize the number of aggregated cells obtained at any stage of the expansion.

The term maximization suggests the possible use of linear programming (LP). While an LP would ensure maximization, this would not be a trivial problem to solve, i.e., the LP relaxation would almost certainly fractionate cells, using their inventory in more than

one aggregated cell. This is not allowed since a cell may be assigned intact to only one aggregated cell. Thus an integer LP would be required which would typically contain 500 or more integer variables. This method would not be expedient in terms of computer usage, especially considering the potential number of integer LPs that may have to be solved for a single estimation cycle.

While we are trying to maximize the number of aggregated cells, it would be satisfactory to obtain close to the maximum if we could preclude the expense in computer time required by an integer LP. For this reason, a heuristic "greedy" algorithm was developed. Complete descriptions of the heuristic algorithm and the LP formulation are contained in Appendix A. The performance of this heuristic is discussed along with the results of the empirical Bayes methods in Chapter IV.

III. ESTIMATION METHODS

A. GENERAL

Once the cell aggregation phase is completed, we begin the attrition rate estimation process. The following notation is used to define the cell data

$$\begin{aligned} K &= \text{number of cells} \\ T &= \text{number of years of data.} \end{aligned} \tag{1}$$

Then for $i = 1, \ldots, K$ and $t = 1, \ldots, T$

$$\begin{aligned} N_i(t) &= \text{inventory of cell } i \text{ in year } t \\ Y_i(t) &= \text{number of attritions in cell } i \text{ in year } t. \end{aligned} \tag{2}$$

The cell data is assumed to be independent binomial, i.e., $Y_i(t) \sim Bin(N_i(t), p_i)$. A success is defined to be an attrition, i.e., an officer from that cell leaves the service during the year. The empirical attrition rate for cell i is given by the Maximum Likelihood Estimator (MLE)

$$\hat{p}_i = \frac{\sum_t Y_i(t)}{\sum_t N_i(t)} . \tag{3}$$

This estimate of p works well for cells with large inventory, but not those with small inventory, which is most often the case in our application.

The MLE has been shown to be dominated by shrinkage methods for $K \geq 3$ (Carter and Rolph, 1974; Efron and Morris, 1975; Casella, 1985). These methods find a grand mean or central attrition rate for the group of cells and a shrinkage factor for each cell. Previous theses have primarily used a common shrinkage factor for all cells; we now allow this shrinkage factor to vary from cell to cell. Each cell's MLE is shrunk towards the central rate by its shrinkage factor. In this way, attrition information from one cell "spills over" into other cells.

The shrinkage methods are developed under the theoretical assumption that the data is normally distributed. Most of the previous studies using empirical Bayes methods have used independent normal data with constant variance (Efron and Morris, 1972,

1973, 1975; Dickinson, 1988). Some applications have used binomial data, using a transformation to make it behave more like normal data (Carter and Rolph, 1974; Efron and Morris, 1975). In Carter and Rolph's estimation of fire alarm probabilities, transformation of the binomial data did not have a large effect on the results (Carter and Rolph, 1974). Our application allows us to investigate the impact of the transformation when applied with more extreme values of p.

Six variations of the empirical Bayes method are applied to the attrition rate estimation problem. The first four are similar in that they use the same iterative procedure to compute the amount of shrinkage for each cell. Of these four, two are on the transformed scale and two on the original scale. Each scale includes two methods of computing the cell variance: one method where the variance is time dependent and the other where it is time independent. The two variance calculations, if they produce like results, provide supporting evidence for the assumption that the data is independent and identically distributed over time. This assumption is certainly questionable, since an officer who remains in a given MOS will move through the YCS and grade cell structure in a predictable manner. As a result, variance that is constant in time (time independent) may not perform as well as one that allows for time variation. The fifth method uses a different iterative procedure to determine the amount of shrinkage and is addressed separately in paragraph III.C.. The final method breaks the cell data into its vector components (service component or commissioning source) before shrinkage techniques are applied and is addressed in paragraph III.D..

B. EMPIRICAL BAYES

1. Transformed Scale

We begin our application of empirical Bayes methods on the transformed scale in an effort to overcome some of the weaknesses in our assumptions. The transformation we use is the Freeman-Tukey transform, a modification of the basic arcsin transformation for binomial data. Its purpose is to stabilize the variance at one and make the data behave more like normal random variables. The form used is

$$X_i(t) = \frac{1}{2}\sqrt{N_i(t)+.5}\left\{arcsin\left(\frac{2Y_i(t)}{N_i(t)+1}-1\right)+arcsin\left(2\frac{Y_i(t)+1}{N_i(t)+1}-1\right)\right\}. \quad (4)$$

Now, let

$$XT_i(t) = \frac{X_i(t)}{\sqrt{N_i(t) + .5}} \qquad \text{for} \quad t = 1, ..., T_i \tag{5}$$

except when $N_i(t) = 0$ (no inventory in year t), in which case $XT_i(t)$ does not exist and we reduce T_i by one. The time average of the transformed values for cell i becomes

$$XTB_i = \frac{1}{T_i} \sum_t XT_i(t) \,. \tag{6}$$

We now need to compute the variance of these time averages. Two methods are used: the first calculation is time dependent, i.e., the variance changes over time, the second is time independent.

The transform stabilizes the variance at one for large values of n and non-extreme values of p. These requirements on n and p are often violated in our application, therefore we have many combinations of n and p for which the variance is less than one. Dickinson was able to discover an interpolative formula which provides a good approximation for the variance of the transformed values, $X_i(t)$, for small values of n and p, and $K \geq 3$ (Dickinson, 1988, pp.8-11). This variance is given by

$$Var(X_i(t)) = \min\{1, V(X_i(t))\} \tag{7}$$

where $V(X_i(t))$ is found by solving

$$V(X_i(t)) = a\,(X_i(t) + C)^{b_1}\,(X_i(t) + C - 1)^{b_2} \tag{8}$$

with

$$C = \sqrt{N_i(t) + .5}\left(\frac{\pi}{2}\right) \tag{9}$$

and

$$a = 1.6835 \qquad b_1 = -.8934 \qquad b_2 = .9881 \,. \tag{10}$$

Equation (8) obviously breaks down if $X_i(t) + C < 1$. When this occurs, we set $X_i(t) + C = 1.001$ and continue. The effect is to use a small but positive variance. The

value of one in Equation (7) dominates for about $X_i(t) + C \geq 2.2$. The variance of the time average is then

$$Var(XTB_i) = \frac{1}{T_i^2} \sum_t Var(XT_i(t)) = \frac{1}{T_i^2} \sum_t \frac{Var(X_i(t))}{N_i(t) + .5} . \tag{11}$$

The second method of computing the variance is the more familiar one. Continuing from Equation (6), the variance of the transformed values is given by

$$Var(XT_i) = \frac{1}{T_i - 1} \sum_t (XT_i(t) - XTB_i)^2 . \tag{12}$$

The variance of their average is therefore

$$Var(XTB_i) = \frac{1}{T_i} Var(XT_i) . \tag{13}$$

Regardless of which variance calculation we use, the same iterative algorithm is used to determine the empirical Bayes estimate for each cell. This estimate, XEB_i, is found by solving

$$XEB_i = \frac{A}{A + Var(XTB_i)} XTB_i + \frac{Var(XTB_i)}{A + Var(XTB_i)} XBB \tag{14}$$

where XEB_i, XTB_i and $Var(XTB_i)$ are cell specific, XBB is the (weighted) grand mean or central attrition rate, and A is the variance of the prior distribution of the cell means. These latter two values must be estimated simultaneously using the following iterative algorithm.

We initialize the algorithm with $A = 0$ and store the previous value of A by

$$A_0 \leftarrow A . \tag{15}$$

Now compute the (weighted) grand mean, XBB. Let

$$\alpha_i = \frac{1}{A + Var(XTB_i)} \tag{16}$$

and

$$\gamma_i = \frac{\alpha_i}{\sum_{j=1}^{K} \alpha_j} . \tag{17}$$

Then

$$XBB = \sum_{i=1}^{K} \gamma_i XTB_i . \tag{18}$$

The updated value of A becomes

$$A \leftarrow A - \frac{K - 1 - \sum_{i=1}^{K} \alpha_i (XTB_i - XBB)^2}{\sum_{i=1}^{K} \alpha_i^2 (XTB_i - XBB)^2} . \tag{19}$$

If $A \leq 0$, set $A = 0$ and exit. This represents the case when there is 100% shrinkage toward the grand mean. If $A > 0$, then check $| A - A_0 | < \varepsilon$ (e.g., $\varepsilon = .0001$). If false, return to Equation (15) for another iteration. If true, the iterations have converged. Exit with the current values of A and XBB for use in Equation (14) to solve for the XEB_i.

Close study of Equation (14) shows that the amount of shrinkage changes from cell to cell since the variance terms are generally not equal. Specifically, cells with higher variance are shrunk more than those with lower variance. In addition, if A is small the shrinkage is greater towards XBB. As $A \rightarrow \infty$, the shrinkage is minimal and the individual cell means dominate.

Once the XEB_i are determined, these values must be transformed back to the original scale. We use

$$\hat{p}_i = \frac{1}{2} \{1 + \sin(XEB_i)\} . \tag{20}$$

2. Original Scale

We return to the assumption of binomial data for original scale calculations. As in the transformed scale, two methods to calculate the variance are used. We begin

with

$$XT_i(t) = \hat{p}_i(t) = \frac{Y_i(t)}{N_i(t)} . \tag{21}$$

As before, if $N_i(t) = 0$ (no inventory in year t), $XT_i(t)$ does not exist and we reduce T_i by one. This leads to the time average for cell i as

$$XTB_i = \frac{1}{T_i} \sum_t XT_i(t) = \frac{1}{T_i} \sum_t \hat{p}_i(t) . \tag{22}$$

The variance calculation which is time dependent, i.e., changes over time, is given by

$$Var(XTB_i) = \frac{1}{T_i^2} \sum_t Var(XT_i(t)) = \frac{1}{T_i^2} \sum_t \frac{\hat{p}_i(t)(1 - \hat{p}_i(t))}{N_i(t)} . \tag{23}$$

We return to Equation (15) with these variance values to perform the iterative algorithm for finding the empirical Bayes estimate, XEB_i, given by Equation (14). Since we are already in the original scale, the transformation given in Equation (20) is ignored, i.e., $\hat{p}_i = XEB_i$.

A problem arises while performing the iterations if a cell has $Y_i(t) = 0 \;\; \forall\, t$ (zero attrition for every year). In this case, the variance given by Equation (23) equals zero. When this value is used in Equation (16), the formula for α_i becomes undefined. We resolve this problem using the Laplace Law of Succession. Assume that $Y_i(t) \sim Bin(N_i(t), p_i)$ and let

$$p_i^* = \frac{Y_i(t) + 1}{N_i(t) + 1} \quad \text{and} \quad q_i^* = \frac{N_i(t) - Y_i(t)}{N_i(t) + 1} \tag{24}$$

be the estimates as prescribed by this law, i.e., Bayes estimator using uniform prior. Then

$$Var\left(\frac{Y_i(t)}{N_i(t)}\right) = \frac{p_i^* q_i^*}{N_i(t)} = \frac{\left(\frac{Y_i(t) + 1}{N_i(t) + 1}\right)\left(\frac{N_i(t) - Y_i(t)}{N_i(t) + 1}\right)}{N_i(t)} . \tag{25}$$

If $Y_i(t) = 0$, then

$$Var\left(\frac{Y_i(t)}{N_i(t)}\right) = \frac{\left(\frac{1}{N_i(t)+1}\right)\left(\frac{N_i(t)}{N_i(t)+1}\right)}{N_i(t)} = \frac{1}{(N_i(t)+1)^2} . \tag{26}$$

This value is used as the summand in Equation (23) whenever $Y_i(t) = 0$ (zero attrition in any year).

For comparison purposes we again compute an alternate variance which is time independent. Continuing from Equation (22), let

$$\tilde{p}_i = \frac{\sum_t Y_i(t)}{\sum_t N_i(t)} . \tag{27}$$

The alternate variance is given by

$$Var(XTB_i) = \frac{1}{T_i^2}\sum_t Var(XT_i(t)) = \frac{\tilde{p}_i(1-\tilde{p}_i)}{T_i^2}\sum_t \frac{1}{N_i(t)} . \tag{28}$$

The problem with cells that have $Y_i(t) = 0 \;\; \forall\, t$ (zero attrition for every year) also occurs here, since the variance given by Equation (28) would equal zero. Using the same concept as before, we obtain the formula

$$Var(XTB_i) = \frac{\sum_t N_i(t)}{\left(1+\sum_t N_i(t)\right)^2}\frac{1}{T_i^2}\sum_t \frac{1}{N_i(t)} . \tag{29}$$

However in this case, this variance formula is necessary only if all years have zero attrition.

As before, we return to Equation (15) to perform the iterative algorithm for finding the empirical Bayes estimate, XEB_i, given by Equation (14).

C. EFRON-MORRIS METHOD

This method is a modification of the iterative algorithm used to estimate A and XBB given by Efron and Morris (Efron and Morris, 1973, pp.127-129). It differs from the method given by Equations (14) through (19) in that it allows the variance of the prior, A, to change from cell to cell. It also gives greater weight to the cells with low variance, and reduces to the James-Stein estimator when the cell variances are constant.

Only one scenario for this method is considered, corresponding to the initial transformed scale, time dependent variance case. Thus, Equations (4) through (11) are repeated, and we begin from the point where we are entering the iterative algorithm. To simplify the following equations, let $D_i = Var(XTB_i)$ as given by Equation (11).

We initialize the algorithm with $A_i = 0$ and $SP_i = 0$ (previous values of S) for $i = 1, \ldots, K$. Let

$$\alpha_i = \frac{1}{A_i + D_i} \tag{30}$$

and

$$\gamma_i = \frac{\alpha_i}{\sum_{j=1}^{K} \alpha_j} . \tag{31}$$

Then

$$\hat{X} = \sum_{i=1}^{K} \gamma_i XTB_i \tag{32}$$

and

$$S_i = (XTB_i - \hat{X})^2 . \tag{33}$$

Now set $i = 1$ and let

$$SN_i = \sum_{j \neq i} \frac{S_j - D_j}{(A_j + D_j)^2} \tag{34}$$

and

$$SD_i = \sum_{j \neq i} \frac{1}{(A_j + D_j)^2} . \tag{35}$$

We then use the Newton-Raphson iteration method to solve

$$A_i = \frac{(S_i - 3D_i) + (A_i + D_i)^2 SN_i}{3 + (A_i + D_i)^2 SD_i} = g(A_i) . \tag{36}$$

First set $AP_i \leftarrow A_i$ for $i = 1, ..., K$ (previous values of A_i). The updated value for A_i becomes

$$A_i \leftarrow A_i - \frac{A_i - g(A_i)}{1 - g'(A_i)} . \tag{37}$$

If $A_i \leq 0$, set $A_i = 0$, let $i = i + 1$, and return to Equation (34). If $A_i > 0$, then test $|A_i - AP_i| > \varepsilon$. If true, return to Equation (36). If false, let $i = i + 1$ and return to Equation (34).

In either case after incrementing i, if $i = K + 1$, we exit and test $|S_i - SP_i| < \varepsilon \ \forall\ i$. If false, we update $SP_i \leftarrow S_i$ and return to Equation (30) with updated values of A_i. If true, the iterations have converged and we must finalize the estimators, XEM_i. Let

$$d_i^* = 3 + (A_i + D_i)^2 \sum_{j \neq i} \frac{1}{(A_j + D_j)^2} \tag{38}$$

and

$$B_i = \left(1 - \frac{4}{d_i^*}\right) \frac{D_i}{A_i + D_i} . \tag{39}$$

If $B_i > 1$, set $B_i = 1$, or if $B_i < 0$, set $B_i = 0$. Then

$$XEM_i = \hat{X} + (1 - B_i)(XTB_i - \hat{X}) . \tag{40}$$

This equation is comparable to Equation (14), which was used to determine the transformed scale estimates, XEB_i, using the previous iterative algorithm. The quantity B_i is

the amount of shrinkage toward the grand mean, $\hat{X}$. The corresponding quantity in Equation (14) is $\dfrac{Var(XTB_i)}{A + Var(XTB_i)}$.

To obtain $\hat{p}_i$, the XEM_i must be transformed back to the original scale using XEM_i in place of XEB_i in Equation (20).

D. VECTOR METHOD

This method is similar to the previous methods in the sense that it uses the aggregated cells produced to meet the defined threshold levels. However, prior to the estimation process, we now partition each aggregated cell by either service component or commissioning source, thus obtaining cells whose elements are vectors. The procedure given by Efron and Morris, modified to compensate for the assumed variance of the time averages, provides the framework for this method (Efron and Morris, 1972, pp.341-344).

The separation by service component or commissioning source requires us to define a third index: the components of the vector. Let

$$P \;=\; \text{number of service components/commissioning sources.} \tag{41}$$

Then for $i = 1, ..., K$, $j = 1, ..., P$ and $t = 1, ..., T$

$$\begin{aligned} N_{ij}(t) &= \text{inventory of cell } i \text{ and vector component } j \text{ in year } t \\ Y_{ij}(t) &= \text{number of attritions in cell } i \text{ and vector component } j \text{ in year } t\,. \end{aligned} \tag{42}$$

Before we had K cells with scalar information, but now we need a $K \times P$ matrix where

$$\sum_{j=1}^{P} N_{ij}(t) \;=\; N_i(t) \quad \text{and} \quad \sum_{j=1}^{P} Y_{ij}(t) \;=\; Y_i(t)\,. \tag{43}$$

A requirement for this method is that $K > P + 2$, for reasons that will soon become obvious.

We begin by defining $X_{ij}(t)$ as the transformed value for $N_{ij}(t)$ and $Y_{ij}(t)$ as given by Equation (4). Continuing in similar manner as the previous transformed scale methods, let

$$XT_{ij}(t) \;=\; \frac{X_{ij}(t)}{\sqrt{N_{ij}(t) + .5}} \qquad \text{for} \quad t = 1, ..., T_{ij} \tag{44}$$

except when $N_{ij}(t) = 0$, in which case $XT_{ij}(t)$ does not exist and we reduce T_{ij} by one.

The time averages of the transformed values are then

$$XTB_{ij} = \frac{1}{T_{ij}} \sum_{t} XT_{ij}(t) . \tag{45}$$

Here we obtain a vector of grand mean values, with each of the P grand means defined by

$$XBB_j = \frac{1}{K} \sum_{i=1}^{K} XTB_{ij} . \tag{46}$$

The transformed scale estimate, δ_{ji}, is then found by solving

$$\delta_{ji} = XBB_j + \{I_P - (K - P - 2)\tilde{S}^{-1}\}(XTB_{ji} - XBB_j) \tag{47}$$

where I_P is the identity matrix of order P, and $\tilde{S}^{-1}$ is found as discussed below. Reversal of the ij index in this and subsequent equations simply means the transpose of the $K \times P$ matrix to a $P \times K$ matrix.

To solve for $\tilde{S}^{-1}$, we begin by defining

$$\tilde{S} = \mathrm{X}_{ji} \mathrm{X}_{ji}^T \tag{48}$$

where

$$\mathrm{X}_{ji} = (XTB_{ji} - XBB_j)\sqrt{V_{ji}} . \tag{49}$$

The V_{ji} matrix is the modification required by our application. (The multiplication in Equation (49) is element-wise as opposed to normal matrix multiplication.) The Efron and Morris method was developed under the assumption that $XTB_{ij} \sim N(\theta_{ij}, 1)$, whereas we are using

$$XTB_{ij} \sim N\left(\theta_{ij}, \frac{1}{T_{ij}^2} \sum_{t} \frac{1}{N_{ij}(t) + .5}\right) \tag{50}$$

provided that $XT_{ij}(t)$ has variance of one. Therefore

$$V_{ij} = \frac{1}{T_{ij}^2} \sum_t \frac{1}{N_{ij}(t) + .5} . \tag{51}$$

We use the requirement that the $P \times P$ matrix resulting from the operations within the brackets in Equation (47) must be nonnegative definite to solve for $\tilde{S}^{-1}$ without having to actually compute its inverse. We proceed by doing an eigenanalysis of $\tilde{S}$, which is seen be Equation (48) to be a real symmetric matrix. We form the diagonal matrix E, which has the eigenvalues, e_j, as its diagonal elements, and the matrix Γ, which has the corresponding eigenvectors as its columns. For any $e_j < (K - P - 2)$, we replace it with the value $(K - P - 2)$. The eigenanalysis provides us with the solution to

$$\begin{aligned} \tilde{S}\,\Gamma_j &= \Gamma_j\, e_j \\ \text{or} \quad \tilde{S}\,\Gamma &= \Gamma\, \mathrm{E} . \end{aligned} \tag{52}$$

Post-multiplying by Γ^T, we obtain

$$\tilde{S} = \Gamma\, \mathrm{E}\, \Gamma^T \tag{53}$$

since Γ is ortho-normal and therefore $\Gamma\,\Gamma^T = I_P$. We then have

$$\tilde{S}^{-1} = (\Gamma\, \mathrm{E}\, \Gamma^T)^{-1} = \Gamma\, \mathrm{E}^{-1}\, \Gamma^T \tag{54}$$

which is easily solved since E^{-1} is found by replacing the diagonal elements of E by their reciprocal. This solution for $\tilde{S}^{-1}$ is then used in Equation (47) to solve for the transformed scale estimates, δ_{ji}. To obtain the attrition rate estimate for a cell, $\hat{p}_{ji}$, we use the inversion formula given by Equation (20) with δ_{ji} in place of XEB_i.

IV. CROSS VALIDATION

A. GENERAL

The six estimation methods discussed in Chapter III are evaluated using cross validation of the data base. This consists of successively holding out one year's data while the other nine years are used to estimate that year's attrition rates. Three measures of effectiveness (MOEs) are used to evaluate the validity of our assumptions and the performance of the estimation methods. Two of these are original scale MOEs--mean absolute deviation (MAD) and chi square statistic. The third is a transformed scale MOE--mean squared error (MSE). Test cases are chosen as input. The results of the cross validation are then discussed.

B. MEASURES OF EFFECTIVENESS

1. Mean Absolute Deviation

The MAD is probably the most useful MOE to the manpower planner. Our version is augmented to display overestimation and underestimation information. Along with the MAD we observe the magnitude of our errors in both directions, which is especially important since the cost of overestimating is generally not the same as the cost of underestimating. While it does not provide a specific value or standard to gauge the performance of our estimation methods, it does provide very useful insight into tendencies to consistently underestimate or overestimate.

For comparison of the estimation methods, we desire a MAD measure that does not depend upon cell inventories, yet still displays the overage/underage information. For these reasons, we use the attrition rate estimates, $\hat{p}_i$, as opposed to the estimated number of attritions, $(\hat{p}_i \bullet N_i(t))$ (where $t =$ validation year), in our MAD calculations. For those estimates obtained in the transformed scale, the XEB_i are inverted back to the original scale using Equation (20) prior to calculating this MOE.

We define the empirical attrition rate for cell i in validation year t as

$$p_i^a = \frac{Y_i(t)}{N_i(t)} \tag{55}$$

except when $N_i(t) = 0$ (no inventory in cell i for the validation year). In this case we do not compute the cell's deviation from the estimated attrition rate since it would

artificially create an overage situation. Therefore, we reduce K by one and continue with the remaining cells (the reduced value of K is then used in the following formulas).

The MAD measures generated for each validation year are

$$\frac{K_u}{K} = \text{fraction of cells with underage} \tag{56}$$

where K_u is the number of cells which have underage,

$$\frac{\sum_i (p_i^a - \hat{p}_i)^+}{\sum_i |p_i^a - \hat{p}_i|} = \text{fraction of MAD due to underage} \tag{57}$$

and

$$MAD = \frac{1}{K} \sum_i |p_i^a - \hat{p}_i| . \tag{58}$$

We also calculate the average MAD over the validation years. Here we use a weighted average, since the number of cells may have been different in some validation years, i.e., a reduced value of K was used in these years. The weighted average takes the form

$$Avg\ MAD = \frac{\sum_t K_t\, MAD_t}{\sum_t K_t} \tag{59}$$

where K_t is the (possibly reduced) number of cells used in validation year t.

2. Chi Square

The chi square test is used as an indicator of how well the binomial model serves as a description of the attrition process. The test statistic is

$$X^2_{(K)}(t) = \sum_i \frac{(Y_i(t) - \hat{p}_i N_i(t))^2}{N_i(t)\, \hat{p}_i (1 - \hat{p}_i)} \tag{60}$$

where t is the validation year. As with the MAD calculations, if $N_i(t) = 0$ we reduce K by one and continue. Additionally, if $\hat{p}_i = 0$ or 1, the denominator equals zero and the summand is undefined. The same course of action is used if this occurs--reduce K by one and continue. Those estimates obtained in the transformed scale are inverted back to the original scale prior to using Equation (60).

This MOE can be used as a gauge. The chi square statistic given by Equation (60) has expected value K and variance $2K$. We are looking for a X^2 value that is less than two standard deviations to the right of the mean, or

$$X^2_{(K)} \leq K + 2\sqrt{2K} . \tag{61}$$

A weighted average chi square is computed in the same manner as the weighted average MAD in Equation (59). However, if the number of cells and thus the degrees of freedom, K, are different over the validation years a problem arises in determining the degrees of freedom for the weighted average. We solve this dilemma by assuming that the weighted average chi square has the original value of K degrees of freedom.

3. Mean Squared Error

The MSE is used to check the validity of our theoretical basis. It is the average squared deviation of the estimated rate from the actual rate, both rates on the transformed scale. The actual rate used is the transformed validation year data. The MSE is defined as

$$L(\delta, \mu) = \frac{1}{K} \sum_i (\delta_i - \mu_i)^2 \tag{62}$$

where

$$\begin{aligned} \delta_i &= XEB_i \\ \mu_i &= XT_i(t) \qquad (t = \text{validation year}) . \end{aligned} \tag{63}$$

Again, if $N_i(t) = 0$, we reduce K by one and continue. A weighted average MSE is also computed similar to Equation (59).

The MSE also has a standard to gauge our model. Using Equations (5) and (12) we can compute a baseline variance for any given validation cell. The MSE for that cell, when compared to the baseline value, provides a figure to gauge the value of using shrinkage estimators instead of the cell averages, XTB_i. There is considerable variability

in these ratios, ranging from 20% to 100%, but 80% appears to be a fair median figure. For example, for cell variances computed from Equation (12) running about 0.15, the MSE hovers about 0.12.

4. Vector Method MOEs

The MOEs discussed above require slight modification before being applied to the vector method described in paragraph III.D.. Recall that this method uses K cells with service component or commissioning source broken out into a vector of length P. An attrition rate estimate, δ_{ij}, is obtained for each of the $K \times P$ matrix components. Thus now we have KP estimates which are compared to the corresponding empirical rates for the validation year. Equations (55) through (63) are modified by replacing all i subscripts with ij, replacing all summations over i by double summations over i and j, and replacing all instances of K by the product KP.

C. TEST CASES

The selection of test cases takes on great importance since they provide the foundation for comparison of these methods. It would be impossible to test every permutation of input parameters; therefore we seek a representative fraction of these which would give an accurate account of the performance of our aggregation and estimation methods. Because we are using a different data base from previous theses on estimation methods, no attempt to duplicate their test cases was made.

An approach based upon Latin square experimental design principles was used to select 30 test cases for the first five estimation methods. The test cases for the vector estimation method are addressed later. In determining the test cases, we randomized when possible and intervened to force pairings only when necessary. To begin, we selected values for the input parameters--T_0, K_0, grade, YCS and MOS. Service component and commissioning source are ignored for these test cases, i.e., all classifications of both are accepted.

To ensure proper representation from small MOS groups one through ten, one MOS from each group was randomly selected: 0302, 1802, 2502, 4002, 3060, 7204, 7545, 7523, 7557 and 7563. Since YCS and grade are strongly correlated, these parameters were selected jointly. To ensure each YCS range within the bounded YCS groups was represented along with a fair representation of grades, four grade/YCS pairs were selected: 1Lt/4 YCS, Capt/7 YCS, LtCol/20 YCS and LtCol(failed select)/26 YCS. The two threshold parameters were also selected jointly, resulting in ten pairs (T_0/K_0): 30.0/30, 20.0/30, 20.0/20, 10.0/30, 10.0/20, 10.0/10, 5.0/30, 5.0/20, 5.0/10 and 5.0/5.

With these choices in place, it was necessary to combine them to define the actual test cases. It was decided to limit the grade/YCS pairs for these cases to 1Lt/4 YCS and Capt/7 YCS due to the large values for the first five threshold pairs. With ten MOSs specified, we sought ten test cases. Thus, each of the first five threshold pairs was listed twice. Each of the four aviation MOSs was randomly assigned to one of the five threshold pairs; the six ground MOSs were then randomly assigned to the remaining pairs. The two grade/YCS pairs were then randomly assigned within a set of common threshold pairs, e.g., for the two cases with T_0 /K_0 of 30.0/30, one was randomly assigned 1Lt/4 YCS, the other was then assigned Capt/7 YCS.

All four grade/YCS pairs would be used with the five remaining threshold pairs. Thus 20 more test cases were generated, with each of the five threshold pairs listed four times. Each MOS was randomly assigned to two distinct threshold pairs, ensuring that the large and major MOS groups were evenly spread throughout the pairs. The four grade/YCS pairs were assigned in random order to each set of common threshold pairs, ensuring that they were evenly spread across large and major MOS groups. The 30 test cases are summarized in Table 3.

The input parameters for six vector test cases were selected from the 30 test cases: Nos. 2, 3, 6, 10, 11 and 20. A small number of vector test cases was initially chosen to investigate the possible advantages of the vector method. If this method appeared to be favorable, then further testing would be conducted.

The six test cases contain a cross section of the input parameters. They include three ground and three aviation MOSs, and use each of the first three grade/YCS pairs twice. The grade/YCS pair of LtCol(FS)/26 YCS was not used because of its extremely low inventory numbers, which when broken out into a vector would have been of little exploratory use. These cases also include six different T_0 /K_0 pairs.

Each of the vector test cases is used twice: first with service component and then with commissioning source as the vector component. All three service components--regular, augmented regular and reserve--were used as vector components. Rather than use all 15 commissioning sources (these 15 are listed below Table 4) as vector components (many of them would contain little or no inventory) five commissioning sources for the ground test cases and five for the aviation test cases were chosen. These five were determined to be the sources which contain the largest percentage of inventory for the respective ground or aviation MOSs. The specific commissioning sources selected along with the other vector test case input parameters are summarized in Table 4.

Table 3. TEST CASES FOR METHODS 1-5

No.	T_0	K_0	MOS	S : L : M	YCS	Grade
1	30.0	30	0302	1 : 1 : 1	4	1Lt
2	30.0	30	7523	8 : 3 : 2	7	Capt
3	20.0	30	3060	5 : 2 : 1	7	Capt
4	20.0	30	7563	10 : 4 : 2	4	1Lt
5	20.0	20	2502	3 : 2 : 1	7	Capt
6	20.0	20	7557	9 : 4 : 2	4	1Lt
7	10.0	30	7204	6 : 2 : 1	4	1Lt
8	10.0	30	1802	2 : 1 : 1	7	Capt
9	10.0	20	7545	7 : 3 : 2	7	Capt
10	10.0	20	4002	4 : 2 : 1	4	1Lt
11	10.0	10	2502	3 : 2 : 1	20	LtCol
12	10.0	10	7557	9 : 4 : 2	26	LtCol(FS)
13	10.0	10	7545	7 : 3 : 2	7	Capt
14	10.0	10	0302	1 : 1 : 1	4	1Lt
15	5.0	30	4002	4 : 2 : 1	4	1Lt
16	5.0	30	0302	1 : 1 : 1	20	LtCol
17	5.0	30	7204	6 : 2 : 1	26	LtCol(FS)
18	5.0	30	7563	10 : 4 : 2	7	Capt
19	5.0	20	3060	5 : 2 : 1	7	Capt
20	5.0	20	7545	7 : 3 : 2	20	LtCol
21	5.0	20	1802	2 : 1 : 1	26	LtCol(FS)
22	5.0	20	7563	10 : 4 : 2	4	1Lt
23	5.0	10	7204	6 : 2 : 1	20	LtCol
24	5.0	10	4002	4 : 2 : 1	26	LtCol(FS)
25	5.0	10	7523	8 : 3 : 2	4	1Lt
26	5.0	10	1802	2 : 1 : 1	7	Capt
27	5.0	5	2502	3 : 2 : 1	7	Capt
28	5.0	5	7557	9 : 4 : 2	20	LtCol
29	5.0	5	3060	5 : 2 : 1	4	1Lt
30	5.0	5	7523	8 : 3 : 2	26	LtCol(FS)

(S : L : M = Small MOS Group : Large MOS Group : Major MOS Group)

Table 4. TEST CASES FOR VECTOR METHOD

No.	T_0	K_0	MOS	YCS	Grade	Service Comp	Comm Source
2	30.0	30	7523	7	Capt	1 2 3	(all)
2	30.0	30	7523	7	Capt	(all)	1 2 3 5 11
3	20.0	30	3060	7	Capt	1 2 3	(all)
3	20.0	30	3060	7	Capt	(all)	1 3 7 10 11
6	20.0	20	7557	4	1Lt	1 2 3	(all)
6	20.0	20	7557	4	1Lt	(all)	1 2 3 5 11
10	10.0	20	4002	4	1Lt	1 2 3	(all)
10	10.0	20	4002	4	1Lt	(all)	1 3 7 10 11
11	10.0	10	2502	20	LtCol	1 2 3	(all)
11	10.0	10	2502	20	LtCol	(all)	1 3 7 10 11
20	5.0	20	7545	20	LtCol	1 2 3	(all)
20	5.0	20	7545	20	LtCol	(all)	1 2 3 5 11

Service Component:

1 - regular
2 - augmented regular
3 - reserve

Commissioning Sources used:

1 - U.S. Naval Academy
2 - Platoon Leader Class-Aviation
3 - Platoon Leader Class-Ground
5 - Aviation Officer Candidate
7 - Officer Candidate Course-Ground
10 - Enlisted Commissioning Program
11 - NROTC-Scholarship

Commissioning Sources not used:

4 - Platoon Leader Class-Law
6 - Marine Aviation Cadet
8 - Officer Candidate Course-Law
9 - Officer Candidate Course-Women
12 - NROTC-Ground College
13 - NROTC-Aviation College
14 - NESEP
15 - All Other

D. RESULTS

For each test case, we apply the aggregation method to meet the threshold levels and then execute the estimation methods. While the ultimate use of these methods is to obtain an attrition rate estimate for the original cell, inspection of these estimates would be of little value in evaluating and comparing the estimation methods. Thus the output from the program takes the form of the MOEs.

The inclusion of the entire output from every test case would not only be cumbersome but would provide an inadequate means of comparing the methods. Therefore the output is summarized in Tables 5 through 7. They contain the output from the first five estimation methods only; the output from the vector test cases is presented later. Sample output for the six methods is contained in Appendix C.

The results summaries list the test case number, the cell inventory threshold, T_0, and the actual number of cells used, K. The level of expansion required to achieve these parameter levels is also listed. For example, test case one had a cell inventory threshold of 30.0, and 24 aggregated cells were obtained by expanding the small, large and major MOS group by YCSs four and five. The value for K is often different from the threshold number of cells, K_0, listed in Table 3. When K is less than K_0, maximum expansion occurred and the threshold was unattainable. When K is greater than K_0, the expansion was the least amount possible to remain above the thresholds. From these test cases we can see that it is difficult to meet the threshold number of cells exactly.

The results summaries then list the weighted average MOEs for each of the five estimation methods. The first row within each test case contains the MAD values, the second row the chi square values and the third row the MSE values. The maximum desired chi square value as given by Equation (61) is listed in parentheses, e.g., for test case one this value is 37.9. This affords easier comparison of the chi square values for the different methods. The values of MSE for both original scale methods are blank because MSE is a transformed scale MOE only.

Before discussing the results in general, some additional comments about specific test cases are necessary. Test case four could not be executed by the Efron-Morris method. This is because one of the aggregated cells contains zero attrition for all ten years of data. As a result, the iterative algorithm does not converge.

Test case 12 was not possible because not even one aggregated cell meeting the cell inventory threshold was obtained with maximum expansion in major MOS group two. This extremely low inventory problem was generally true for all test cases involving the LtCol(FS)/26 YCS pair. Test cases 17, 21 and 24 obtained only three aggregated cells

with maximum expansion in major MOS group one. Test case 30 was also from major MOS group two, and obtained only one aggregated cell meeting the cell inventory threshold. As a result, test case 30 was changed to Major(FS)/18 YCS so that results for these low thresholds could be obtained.

As the thresholds became low (test cases 23-30) cells with inventory much larger than T_0 were being obtained prior to any aggregation. This was especially true for the 1Lt/4 YCS and Capt/7 YCS pairs. To avoid masking the results of low inventory thresholds by actually using large inventory cells, the service component/commissioning source parameters for test cases 26, 27 and 29 were changed. Rather than accepting all classifications of these parameters, only one classification for each was accepted. Thus test case 26 was executed with regular/USNA, test case 27 was executed with augmented regular/PLC-ground, and test case 29 was executed with regular/NROTC-scholarship.

We now focus our attention on the results of these test cases with respect to the MOEs. The weighted average MAD figures vary little within each test case over the five methods. This suggests that the total deviation from the validation year empirical attrition rate was the same for all methods. However, this figure does not identify whether the deviations were overestimations or underestimations. The fraction of MAD from underage (not listed in the results summaries) was studied to gain more insight into this important consideration. For each of the 29 test cases (no results for test case 12), a weighted average fraction of MAD from underage was computed for each of the five estimation methods (weighted by the number of cells just as the weighted averages for the MOEs). A weighted average of these 29 values was then computed. This overall weighted average indicates the tendency of the method to underestimate or overestimate--averages above 0.5 indicate a tendency to underestimate; averages below 0.5 indicate a tendency to overestimate. The author is unaware of any information comparing the relative costs of underage and overage. Hence, as a default, we look for values of 0.5, which is a balance between overestimation and underestimation. The averages calculated were: TS1 = 0.426; TS2 = 0.436; OS1 = 0.481; OS2 = 0.512; and EM = 0.452. Although the MAD figures were generally the same for all methods, these averages indicate that the tendencies to overestimate or underestimate may not be the same. The original scale methods seem to have achieved more balance than the transformed scale methods.

The chi square results were not entirely consistent across test cases nor across methods within a test case. These results are discussed first by comparisons between test cases; then by comparisons within test cases.

Of the first ten test cases, only three (Nos. 3, 5 and 8) had weighted average chi square values within the acceptable range. These three test cases expanded only into the large MOS group, whereas of the seven test cases which were unacceptable, all but one (No. 10) expanded into the major MOS group. Of the last 20 test cases, only four had chi square values outside the acceptable range (Nos. 13, 14, 15, and 18). Of these four, two expanded into the large MOS group, and two into the major MOS group. All the test cases with unacceptable values had either 1Lt/4 YCS or Capt/7 YCS pairs. They were fairly well spread across MOS groups. None of the test cases with lower thresholds (Nos. 19-30) had unacceptable chi square values. This suggests that lower thresholds, which result in less expansion, achieve more acceptable results with respect to this MOE.

To investigate this claim further, different combinations of threshold levels for test cases seven and nine were used. The results are contained in Table 8. These results reinforce the claim that lower thresholds are in fact better, since in both cases the chi square results improved as the thresholds, and therefore the level of expansion, were reduced. When comparing these extra test cases, keep in mind that the chi square values for each set of threshold values should be compared to the acceptable range for that specific number of cells; comparisons across test cases with different values of K are not valid. An important aspect of the argument for lower thresholds is that the thresholds must be considered jointly. For example, in test case seven, with a T_0/K pair of 10.0/6, the chi square values were nearly acceptable, whereas with 5.0/19 they were clearly unacceptable. Thus we should be most aware of the value $T_0 \times K_0$.

We now turn our attention to comparing the chi square values within a test case. Several test cases had chi square values that varied significantly over the estimation methods (Nos. 4, 6 and 22). These three all had 1Lt/4 YCS pairs and were large MOS group four. In test cases four and six the chi square values for the transformed scale methods were not too much larger than the desired maximum; the values for the original scale methods were significantly larger than the desired maximum. In test case 22 the chi square values for the transformed scale methods were acceptable, however the values for the original scale methods were again significantly larger than the desired maximum.

Several other test cases had varying chi square values, but to a lesser degree. In test case eight, only the transformed scale methods had acceptable chi square values, the original scale methods exceeded the desired maximum, although not by a significant margin. In test case 19, only the OS2 method exceeded the desired maximum. All methods for test cases 16, 25, and 27 were within the desired maximum, however the chi square values varied to a large degree over the five methods. Thus, using the chi square

MOE, it appears that the transformed scale methods were generally the same, and as a group outperformed the original scale methods.

The MSE was used only with the transformed scale methods, and thus no comparison with original scale methods can be made. The values for this MOE were generally equal between methods within a test case, and acceptable overall.

The results for the vector method test cases are summarized in Table 9. The table lists the value $K \times P$ (instead of K) because this is the number of estimates obtained and compared to the validation year empirical attrition rates with this method.

Test case two with commissioning source as the vector component had to be modified because only seven aggregated cells were obtained. As a result, $K = P + 2$, and the vector method could not be conducted. Therefore, commissioning source three (Platoon Leader Class-ground) was deleted as a vector component and the test case run with only four commissioning sources (1, 2, 5 and 11). Test case 20 with service component as the vector component was infeasible. By starting with a low cell inventory threshold (5.0), when the cells were broken out into the vector components their inventory became extremely low. As a result, when validating year five, two of the cells had zero inventory for all of the remaining nine years for service component three (reserve). Thus the value for XTB_{ij} , given by Equation (45), becomes undefined and the method cannot be completed.

The results of the vector method with respect to the MOEs is similar to the results of the previous five estimation methods. This method produced acceptable MAD and MSE values, but its chi square values were fairly inconsistent. Test cases two and ten had unacceptable chi square values for both vector components; test case six had an unacceptable chi square value with service component as the vector component. Recall that these test cases also had unacceptable values with the vector component collapsed.

No fair comparison with the first five estimation methods can be made using the summarized results. Obviously the MAD and MSE quantities will be larger since we are comparing three to five times as many estimates to empirical rates with the vector method. Thus a different evaluation technique must be used.

The vector method is designed to take advantage of any correlation between the cells when broken out into vector components. To see if this is occurring we must look at the matrix $\tilde{S}^{-1}$ as given by Equation (54). In all of the vector test cases, this matrix was essentially diagonal, indicating little correlation between the vector components. In addition, all of the eigenvalues, which become the elements of the diagonal matrix E, were less than $(K - P - 2)$. Therefore, the eigenvalues were replaced by this quantity and the

Table 5. SUMMARY OF RESULTS (CASES 1-10)

No.	T_0	K	Expansion Required	MOE	TS1	TS2	OS1	OS2	EM
1	30.0	24	S:(4,5) L:(4,5) M:(4,5)	MAD X(37.9) MSE	0.102 98.79 0.079	0.101 99.28 0.078	0.102 100.99	0.102 100.96	0.103 101.23 0.080
2	30.0	8	S:(7) L:(7) M:(7)	MAD X(16.0) MSE	0.091 28.16 0.077	0.090 27.68 0.076	0.092 28.71	0.091 28.68	0.091 27.73 0.076
3	20.0	31	S:(1-3,6-19) L:(6-8)	MAD X(46.7) MSE	0.056 37.49 0.062	0.056 37.80 0.062	0.055 39.58	0.055 42.85	0.055 38.38 0.062
4	20.0	23	S:(1-5,8-19) L:(1-5,8-19) M:(1-5,8-19)	MAD X(36.6) MSE	0.029 39.34 0.035	0.029 47.36 0.037	0.026 95.41	0.024 139.14	
5	20.0	20	S:(1-3,6-19) L:(7)	MAD X(32.6) MSE	0.048 20.75 0.048	0.049 20.83 0.049	0.047 24.34	0.047 24.20	0.048 21.41 0.049
6	20.0	22	S:(1-5,8-19) L:(1-5,8-19) M:(3-5)	MAD X(35.3) MSE	0.029 40.42 0.037	0.029 50.70 0.039	0.027 90.93	0.025 127.86	0.029 65.20 0.040
7	10.0	32	S:(4,5) L:(4,5) M:(4)	MAD X(48.0) MSE	0.129 111.67 0.137	0.130 113.23 0.137	0.130 112.64	0.130 113.64	0.133 114.28 0.140
8	10.0	28	S:(1-3,6-19) L:(1-3,6-11)	MAD X(43.0) MSE	0.039 34.37 0.039	0.038 34.70 0.040	0.039 45.16	0.038 49.60	0.039 35.54 0.040
9	10.0	14	S:(7) L:(7) M:(7)	MAD X(24.6) MSE	0.117 38.85 0.130	0.115 37.50 0.127	0.117 39.59	0.117 39.62	0.115 40.26 0.132
10	10.0	27	S:(4.5) L:(4,5)	MAD X(41.7) MSE	0.128 64.33 0.137	0.129 65.92 0.137	0.129 65.62	0.128 66.61	0.127 62.64 0.135

TS1 - Transformed scale, time dependent variance

TS2 - Transformed scale, time independent variance

OS1 - Original scale, time dependent variance

OS2 - Original scale, time independent variance

EM - Efron-Morris method

matrix E always had $(K - P - 2)$ as its diagonal elements. Because these results indicated no worthwhile improvements over the first five methods, no further testing of the vector method was conducted.

Finally, the performance of the heuristic aggregation algorithm listed in Appendix A was also evaluated. For each test case, the total inventory of cells below T_0 was summed and this value divided by T_0. The integer part of this number provides an upper bound on the number of aggregated cells that can be obtained by any aggregation technique. This upper bound was compared to the actual number of aggregated cells produced by the algorithm. The algorithm achieved the maximum in 71.4% (20 of 28) of the test cases. It achieved one less than the maximum in 21.4% (6 of 28) of the test cases, and two less than the maximum in 7.2% (2 of 28) of the test cases (only 28 test cases required aggregation: No. 12 was infeasible; No. 14 all cells were above T_0). This performance is acceptable for our application.

Table 6. SUMMARY OF RESULTS (CASES 11-20)

No.	T_0	K	Expansion Required	MOE	TS1	TS2	OS1	OS2	EM
11	10.0	11	S:(20-25) L:(20-25)	MAD X(20.4) MSE	0.109 18.79 0.141	0.106 19.75 0.142	0.108 19.96	0.107 20.53	0.107 18.99 0.139
12	10.0	0	S:(26) L:(26) M:(26)	MAD X(0.0) MSE					
13	10.0	14	S:(7) L:(7) M:(7)	MAD X(24.6) MSE	0.117 38.85 0.130	0.115 37.50 0.127	0.117 39.59	0.117 39.62	0.115 40.26 0.132
14	10.0	10	S:(4,5) L:(4,5)	MAD X(18.9) MSE	0.113 57.80 0.102	0.110 60.07 0.103	0.113 57.71	0.113 57.80	0.115 59.36 0.104
15	5.0	32	S:(4,5) L:(4,5)	MAD X(48.0) MSE	0.137 69.96 0.156	0.140 70.72 0.158	0.139 70.27	0.139 72.01	0.139 68.15 0.155
16	5.0	32	S:(20-25) L:(20-25) M:(20-22)	MAD X(48.0) MSE	0.122 39.41 0.146	0.122 40.38 0.146	0.121 44.35	0.121 47.84	0.121 42.27 0.148
17	5.0	3	S:(26) L:(26) M:(26)	MAD X(7.9) MSE	0.166 3.66 0.0.189	0.166 3.57 0.188	0.168 3.88	0.168 3.78	0.168 4.02 0.202
18	5.0	29	S:(6,7) L:(6,7) M:(7)	MAD X(44.2) MSE	0.141 79.04 0.176	0.141 78.23 0.176	0.142 78.61	0.141 81.29	0.141 78.84 0.177
19	5.0	24	S:(2,3,6-10)	MAD X(37.9) MSE	0.085 28.95 0.122	0.082 29.28 0.123	0.080 30.91	0.084 43.36	0.080 35.44 0.132
20	5.0	19	S:(20-25) L:(20-25) M:(20)	MAD X(31.3) MSE	0.111 17.29 0.113	0.113 17.29 0.114	0.113 18.92	0.106 23.37	0.110 18.28 0.114

TS1 - Transformed scale, time dependent variance

TS2 - Transformed scale, time independent variance

OS1 - Original scale, time dependent variance

OS2 - Original scale, time independent variance

EM - Efron-Morris method

Table 7. SUMMARY OF RESULTS (CASES 21-30)

No.	T_0	K	Expansion Required	MOE	TS1	TS2	OS1	OS2	EM
21	5.0	3	S:(26) L:(26) M:(26)	MAD X(7.9) MSE	0.167 3.66 0.189	0.166 3.57 0.188	0.168 3.88	0.168 3.78	0.168 4.02 0.202
22	5.0	22	S:(1-5,8-19) L:(3,4)	MAD X(35.3) MSE	0.032 23.56 0.059	0.042 15.03 0.041	0.024 63.10	0.022 92.85	0.032 22.56 0.056
23	5.0	9	S:(20-25) L:(20)	MAD X(17.5) MSE	0.121 8.12 0.141	0.121 8.19 0.141	0.119 8.65	0.121 9.25	0.120 8.17 0.137
24	5.0	3	S:(26) L:(26) M:(26)	MAD X(7.9) MSE	0.166 3.66 0.189	0.166 3.57 0.188	0.168 3.88	0.168 3.78	0.168 4.02 0.202
25	5.0	10	S:(1-6,8-19) L:(1-6)	MAD X(18.9) MSE	0.050 5.09 0.044	0.051 4.78 0.037	0.020 15.35	0.020 16.54	0.048 4.85 0.039
26	5.0	11	S:(1-3,6-19) L:(6,7)	MAD X(20.4) MSE	0.138 13.61 0.184	0.138 13.65 0.186	0.138 14.09	0.135 16.10	0.135 14.28 0.189
27	5.0	6	S:(6-8)	MAD X(12.9) MSE	0.070 4.64 0.075	0.069 4.55 0.074	0.055 6.88	0.050 9.41	0.068 4.60 0.074
28	5.0	7	S:(20,21)	MAD X(14.5) MSE	0.114 7.59 0.140	0.116 7.67 0.143	0.118 7.99	0.113 8.74	0.116 8.37 0.147
29	5.0	5	S:(4,5)	MAD X(11.3) MSE	0.170 6.36 0.211	0.169 6.23 0.210	0.168 7.64	0.167 8.25	0.169 8.06 0.281
30	5.0	6	S:(1-6,8-19) L:(17-19)	MAD X(12.9) MSE	0.112 6.15 0.136	0.110 6.40 0.136	0.110 7.31	0.104 8.73	0.108 6.42 0.135

TS1 - Transformed scale, time dependent variance

TS2 - Transformed scale, time independent variance

OS1 - Original scale, time dependent variance

OS2 - Original scale, time independent variance

EM - Efron-Morris method

Table 8. SUMMARY OF RESULTS (CASES 7 AND 9 EXPANDED)

No.	T_0	K	Expansion Required	MOE	TS1	TS2	OS1	OS2	EM
7-1	10.0	32	S:(4,5) L:(4,5) M:(4)	**X(48.0)**	111.67	113.23	112.64	113.64	114.28
7-2	10.0	17	S:(4,5) L:(4)	**X(28.7)**	45.31	45.99	45.90	46.58	43.68
7-3	10.0	6	S:(4,5)	**X(12.9)**	13.08	12.98	13.41	13.53	13.03
7-4	5.0	19	S:(4,5) L:(4)	**X(31.3)**	49.28	49.53	49.23	50.84	46.91
7-5	5.0	7	S:(4,5)	**X(18.9)**	14.42	14.27	14.82	14.93	14.18
7-6	5.0	4	S:(4)	**X(9.7)**	9.20	8.88	9.22	9.20	9.49
9-1	10.0	14	S:(7) L:(7) M:(7)	**X(24.6)**	38.85	37.50	39.59	39.62	40.26
9-2	10.0	4	S:(7)	**X(9.7)**	10.35	10.14	10.58	10.57	10.11
9-3	5.0	17	S:(7) L:(7) M:(7)	**X(28.7)**	44.95	42.87	44.40	46.03	46.15
9-4	5.0	5	S:(7)	**X(11.3)**	11.91	11.69	12.16	12.15	11.64

TS1 - Transformed scale, time dependent variance

TS2 - Transformed scale, time independent variance

OS1 - Original scale, time dependent variance

OS2 - Original scale, time independent variance

EM - Efron-Morris method

Table 9. SUMMARY OF RESULTS (VECTOR METHOD)

No.	T_0	*KP*	Vector Component	MOE	Vector Method
2	30.0	24	SC	MAD X(37.9) MSE	0.149 61.03 0.229
2	30.0	28	CS	MAD X(43.0) MSE	0.222 115.34 0.398
3	20.0	93	SC	MAD X(120.3) MSE	0.158 110.96 0.219
3	20.0	155	CS	MAD X(190.2) MSE	0.157 132.37 0.201
6	20.0	66	SC	MAD X(89.0) MSE	0.062 109.19 0.074
6	20.0	100	CS	MAD X(128.3) MSE	0.083 58.58 0.081
10	10.0	81	SC	MAD X(106.5) MSE	0.212 251.63 0.368
10	10.0	120	CS	MAD X(151.0) MSE	0.321 707.85 0.625
11	10.0	42	SC	MAD X(60.3) MSE	0.193 44.86 0.214
11	10.0	60	CS	MAD X(81.9) MSE	0.222 53.55 0.225
20	5.0	72	SC	MAD X(0.0) MSE	
20	5.0	100	CS	MAD X(128.3) MSE	0.254 55.92 0.245

SC - service component

CS - commissioning source

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The results indicate that the desired stability in estimating attrition rates for low inventory cells has been achieved with the aggregation and estimation methods presented in this study. The use of "shrinkage" methods applied to well selected groups of cells allows for the achievement of quality estimates of attrition in the face of low inventory numbers for the individual cells.

None of the six estimation methods stood out as a clear favorite. The vector method did not provide any additional benefits using service component or commissioning source as vector components. Since it is a more complicated method and has the potential to become unsolvable with zero inventory vector components, it appears to be the least favorite. Perhaps more success would be obtained with alternative classifications for the vector component.

The Efron-Morris method also involves more computational effort than the first four empirical Bayes methods. Its performance was very much similar to the transformed scale, time dependent variance method since the only difference between them is the iterative algorithm used to determine the amount of shrinkage. The Efron-Morris method has the potential to become unsolvable when a cell has zero attrition for every year--a distinct possibility when dealing with low inventory cells. This suggests that it is the least favorite of the first five methods.

Of the remaining four methods, there seems to be only small difference between the time dependent variance and the time independent variance methods on the same scale. In test cases where the chi square values were marginal or unacceptable, the time dependent variance methods were usually better. In these same test cases, the transformed scale methods performed better than the original scale methods. Therefore, if one method was to be singled out as best, it would be the first method: transformed scale, time dependent variance.

The tendency to overestimate or underestimate as shown by the weighted average fraction of MAD from underage may also be a consideration when selecting a method. An analysis of this type must weigh the costs of overestimating versus the costs of underestimating, which generally are not the same. This type of an analysis is beyond

the scope of this study. In addition, further testing of the methods would be required to gain a more accurate estimate of this tendency.

The threshold levels also seem to strongly influence the performance of the estimation methods. It appears that expansion past the large MOS group begins to detract from homogeneous attrition behavior. While further study would be required to identify optimal threshold levels, it is apparent that both thresholds should not exceed 20.0, and the value of $T_0 \times K_0$ should not exceed 100.

A method for dealing with cells whose inventory is much greater than T_0 must be developed. In some test cases, cells with inventory three or more times as large as T_0 were obtained and used in the estimation process. This did not seem to affect the results, as they were present in almost all test cases. These cells could be disaggregated into multiple cells with inventory closer to the threshold, although the effect of this has not been determined.

B. RECOMMENDATIONS

The proposed aggregation method should be implemented as a method of identifying additional cells to be used in the attrition rate estimation process. This method provides greater homogeneity of attrition behavior among cells over the current method.

The empirical Bayes estimation methods developed in this study are recommended for use in estimating the attrition rates for low personnel inventory cells.

Overall, the empirical Bayes estimation methods when combined with the proposed aggregation method have achieved the stability in attrition rate estimation that is required to provide a foundation for manpower planning.

APPENDIX A. AGGREGATION ALGORITHMS

A. HEURISTIC ALGORITHM

The heuristic algorithm for aggregating cells is as follows:

1. Given a set of cells, S, and the (time average) inventory of each cell, INV_c, partition S into two subsets as follows:
$$S_1 = \{c : c \in S \, ; \, INV_c \geq T_0\}$$
$$S_2 = \{c : c \in S \, ; \, INV_c < T_0\}$$
2. Put the cells in S_1 into the set of aggregated cells, K.
3. Order the cells in S_2 according to size of their inventory:
$$INV_1 \leq INV_2 \leq \cdots \leq INV_n \qquad n = |S_2|$$
4. Start with c_n, the cell in S_2 with the largest inventory. Find the smallest cell in S_2, c^*, that when united with c_n the resulting total inventory will meet or exceed T_0. Combine its data with c_n, put c_n into K, and remove c^* from S_2 (the modified set S_2 will now be referred to as S_2^-). Repeat the procedure with c_{n-1}, and so forth.
5. If no cell in S_2^- when combined with the current largest cell, c_{n-i}, exceeds T_0, use the next largest cell, c_{n-i-1}, and remove c_{n-i-1} from S_2^-. This will create an aggregated cell that is still below threshold. Return to the procedure in Step 4 of trying to find c^*. If no such cell is contained in S_2^-, use c_{n-i-2}, and so forth.
6. Continue this procedure until the sum of all the cells remaining in S_2^- is less than T_0. These cells are sequentially added to the aggregated cells in K in Step 7.
7. Add the largest cell in S_2^- to the smallest cell in K, and update its average inventory. Add the next largest cell in S_2^- to the current smallest cell in K, and update the inventory. Continue until all cells in S_2^- have been used.

We now have $|K|$ aggregated cells which exceed the threshold, T_0, to use in the attrition rate estimation procedure.

B. INTEGER LINEAR PROGRAM

The formulation as an integer linear program is as follows:

<u>Index Use</u>

c	cell (before aggregation)
a	aggregated cell

<u>Given Data</u>

INV_c	average inventory of cell c
T_0	threshold cell inventory

Binary Variables

$X_{c,a}$ 1 indicates use cell c in aggregated cell a

Z_a 1 indicates use cell a for aggregation

Formulation

$$\text{MAX} \quad \sum_a Z_a$$

subject to

$$\sum_a X_{c,a} \le 1 \qquad \forall\, c \qquad \text{(each cell used at most once)}$$

$$\sum_c INV_c \bullet X_{c,a} \ge T_0 \bullet Z_a \qquad \forall\, a \qquad \text{(aggregated cell must have size} \ge T_0\text{)}$$

$$X_{c,a},\ Z_a \ \in \ \{0,1\}$$

APPENDIX B. COMPUTER PROGRAMS

A. GENERAL

A computer program written in FORTRAN is used to conduct the cross validation using the methods developed in this thesis. Although the program consists of 33 subroutines, 6 function subroutines, and over 2000 lines of code, it can be easily summarized by breaking it into the two areas of the thesis: cell aggregation and estimation methods.

The main program and aggregation subroutines (listed in paragraph B) read the input parameters and execute the expansion and aggregation methods discussed in Chapter II. An existing program written by Luis Uribe, an independent contractor under the direction of Professor Read, underwent extensive modification to fulfill these requirements. The input parameters--T_0, K_0 , MOS, YCS, grade, service component(s), and commissioning source(s)--are read by Subroutine GETPAR either in the interactive mode via the terminal or by using MC87 EXEC (listed in paragraph E). Uribe uses an innovative method to estimate the amount of expansion required to meet the threshold parameters. This approach precludes the requirement to read the data base after each step in the expansion process which would be extremely computer time intensive. Inventory information is extracted from the data base and stored in a separate data file for each pay grade (a sample data file and the program used to create it are listed in paragraph F). The data file is accessed via the user's A-disk which is significantly faster than accessing the data base through MVS. Subroutine READET reads the appropriate data file for the specified grade and constructs a table of cells for those records that are in the same major MOS group as the user defined MOS, and meet the service component and commissioning source parameters. All YCSs are accepted, since the extent of expansion is not yet determined. Function NCEVAL screens this table using the current level of expansion and estimates the number of aggregated cells with average inventory greater than or equal to T_0 that will be obtained. If this number is less than K_0, Subroutine EXPAND begins the expansion stages as described in paragraph II.C.. After each increment of expansion, NCEVAL screens the table and estimates the number of aggregated cells that will be obtained. This loop through EXPAND and NCEVAL continues until the estimated number of aggregated cells meets the threshold, K_0. The estimated number of cells and the level of expansion are then displayed on the terminal screen.

The user may elect to go forth and read the data base to determine the actual number of aggregated cells obtained, or may elect to change the level of expansion.

The level of expansion is changed through the variable AGGPCT. This variable estimates the effectiveness of the heuristic aggregation method listed in Appendix A. To estimate the number of aggregated cells that will be obtained, NCEVAL compares the cells which meet the expansion criteria to the minimum inventory threshold, T_0. Those that are greater than T_0 will obviously produce one aggregated cell. The inventory of those that are less than T_0 is summed. The estimated number of cells is then the total of the number of cells greater than T_0 and AGGPCT times the sum of the cell inventory below T_0 divided by T_0. The variable AGGPCT is initially set at 0.9, but can be interactively changed via the terminal. By increasing the value of AGGPCT we can decrease the level of expansion; by decreasing the value of AGGPCT we can increase the level of expansion.

Once we decide to go forth and read the actual data base, Subroutine READER extracts records meeting the expansion criteria developed using EXPAND and NCEVAL and pools them into cells. Subroutine AGGREG aggregates these cells to meet the average inventory threshold, T_0 . The actual number of aggregated cells obtained is then compared to the threshold number of cells, K_0. Again, the user has the option of changing the level of expansion to obtain more or fewer cells, or continuing on to the estimation process.

The first five estimation methods are called by SUBROUTINE MC87BZ (listed in paragraph C). The estimation methods are contained in separate subroutines: EBTS1, EBTS2, EBOS1, EBOS2 and EMTS (EB-empirical Bayes; EM-Efron-Morris; TS-transformed scale; OS-original scale; 1-time dependent variance; 2-time independent variance). The iterations required by the first four methods are conducted in Subroutine EBITER; the Efron-Morris iterations are conducted in Subroutine EMITER. The MOEs are then computed by Subroutines MSE and OSMOE.

If the vector method is to be used, Subroutine BKDOWN then breaks the cells out by their vector components (a vector of length three for service component; a vector of length five for commissioning source). The vector estimation method is contained in Subroutine MC87V (listed in paragraph D). Since all of its computations are unique, this subroutine is self-contained with the exception of the transformation formula, which is contained in Function FTTV.

B. MAIN PROGRAM AND AGGREGATION SUBROUTINES

```
C --- PROGRAM TO CONDUCT AGGREGATION AND ESTIMATION METHODS
C
C --- PARAMETER MXY MUST BE UPDATED TO REFLECT EXACT NO. YEARS OF DATA
C --- PARAMETER MXP IS MAX LENGTH OF 3RD DIMENSION P-VECTOR
C --- PARAMETER MXK IS MAX NUMBER OF AGGREGATED CELLS (MAX NO)
      PARAMETER (MXX=600, MXY=10, MXP=6, MXK=50)
      PARAMETER (NMS=81, NG=14, NLG=6, NMG=4, NYB=4, NYE=18, NYEG=4)
C
      INTEGER ST1,ST2,LYR
      INTEGER SYCS(31), NYCS
      INTEGER SYCSG(31),SYCSL(31),SYCSM(31),NYCSG,NYCSL,NYCSM
      INTEGER SMOS(30), NMOS
      INTEGER SVCMP(5), NSC
      INTEGER SCSRC(16),NCSR
      INTEGER SGRD
      INTEGER*2 MOSGR(2,NMS), YCSB(NYE,NYB,NYEG), VYC(NYE)
      INTEGER*2 LGRP(NG), MGRP(NLG)
      REAL INV(MXX,MXY),Y(MXX,MXY), SINV(MXX,MXY),SY(MXX,MXY)
      INTEGER DATA(MXY)
C --- ARRAYS FOR MC87BZ
      REAL XTB(MXX),VXTB(MXX),XEB(MXX),A(MXX)
C --- ARRAYS FOR MC87V
      REAL XTBJI(MXP,MXK), DELTA(MXP,MXK), X(MXP,MXK)
      REAL XVYR(MXP,MXK), VYRINV(MXP,MXK), VYRY(MXP,MXK)
      REAL BSTAR(MXP,MXP), S(MXP,MXP), GAMMA(MXP,MXP)
      REAL XBBJ(MXP), EVAL(MXP)
C
      INTEGER*2 PTRTBL(MXX, 2), INDX(MXX), MKG(MXX), RETTBL(MXX,3)
      INTEGER*2 PTBL(MXX, 3), BKTBL(MXX,3)
      REAL AVINV(MXX), RETINV(MXX)
      DATA MKG/MXX*0/
C --- ASSIGN MOS TO SMALL, LARGE AND MAJOR MOS GROUP
      DATA MOSGR/013,1, 020,2, 027,2, 038,2, 039,2,
     *   005,3, 007,3, 049,3, 052,3,
     *   074,4, 079,4, 085,4, 101,4,
     *   016,5, 060,5, 064,5, 076,5, 111,5, 116,5,
     *   132,6, 134,6, 135,6, 139,6,
     *   143,7, 147,7, 150,7, 153,7, 154,7, 155,7, 170,7,
     *   149,8, 151,8,
     *   160,9, 161,9, 164,9, 166,9, 167,9, 168,9, 178,9,
     *   173,10, 174,10, 175,10, 176,10, 177,10, 179,10, 144,10,
     *   145,10, 165,10,
     *   001,11, 006,11, 012,11, 015,11, 019,11, 026,11, 037,11,
     *   048,11, 051,11, 059,11, 070,11, 075,11, 078,11, 084,11,
     *   087,11, 100,11, 110,11, 115,11, 131,11, 138,11, 217,11,
     *   172,12, 187,12, 188,12, 189,12,
     *   142,13, 146,13, 148,13, 152,13, 156,13, 163,13, 169,13,
     *   088,14  /
      DATA LGRP/1,1,4*2,3,3,4,4,5,5,5,6/
      DATA MGRP/1,1,2,2,3,4/
C --- CREATE YCS EXPANSION BOUNDS
      DATA YCSB/1,2,3,4,5,6, 8,9,10,11,12,13,14,15,16,17,18,19,
     *          7,17*0,   20,21,22,23,24,25,12*0,   26,17*0,
     *          1,2,3,4,5, 8,9,10,11,12,13,14,15,16,17,18,19,1*0,
```

```
     *          6,7,16*0,   20,21,22,23,24,25,12*0,   26,17*0,
     *          1,2,3,4,5,6, 8,9,10,11,12,13,14,15,16,17,18,19,
     *          7,17*0,   20,21,22,23,24,25,12*0,   26,17*0,
     *          1,2,3, 6,7,8,9,10,11,12,13,14,15,16,17,18,19,1*0,
     *          4,5,16*0,   20,21,22,23,24,25,12*0,   26,17*0    /
C --- INITIALIZE INVENTORY AND ATTRITION ARRAYS
      DO 1 I=1,MXX
       DO 2 J=1,MXY
        SINV(I,J)=0
          SY(I,J)=0
         INV(I,J)=0
           Y(I,J)=0
    2  CONTINUE
    1 CONTINUE
C --- DEFINE FILE FOR OUTPUT
      CALL EXCMS('FILEDEF 11 DISK MC87 OUTPUT A')
C --- FIRST/LAST YEAR OF DATA ON TAPE.  UPDATE WHEN NECESSARY
      ST1=77
      NYR=MXY
      LYR=ST1+NYR-1
C --- INITIAL VALUE FOR AGGREGATION ESTIMATION PERCENTAGE
      AGGPCT=0.9
      ICYCLE=1
C --- GET INPUT PARAMETERS
      CALL GETPAR(AIMIN,NO,NMOS,SMOS,NYCS,SYCS,SGRD,
     *            NSC,SVCMP, NCSR,SCSRC, IGR,MOSGR,NMS, ISFLAG)
C --- MAJOR GROUP IS MG, LARGE GROUP LG, GROUP IGR, YCS BLOCK IY
      LG=LGRP(IGR)
      MG=MGRP(LG)
             WRITE(6,*) ' '
             WRITE(6,*) '---- GR,LG,MG=',IGR,LG,MG
             WRITE(6,*) ' '
C --- READ EVALUATION TABLE. SELECT ONLY RECS PASSING SELECTION CRITERIA
      CALL READET(RETTBL,RETINV,MXX,NRET, SGRD, NSC,SVCMP, NCSR,SCSRC,
     *            MG,LGRP,MGRP, MOSGR,NMS)
  5   RC=0
      IGX=IGR
      LGX=0
      MGX=0
      NYCSG=1
      SYCSG(1)=SYCS(1)
      NYCSL=1
      SYCSL(1)=SYCS(1)
      NYCSM=1
      SYCSM(1)=SYCS(1)
      NCTOT=0
      NC=NCEVAL(AIMIN,IGX,LGX,MGX,NYCSG,SYCSG,RETTBL,RETINV,NRET,MXX,
     *  LGRP,MGRP,NMS,AGGPCT,IGR,LG)
C --- DO WHILE NCTOT<NO & RC=0   (EXPAND AS LONG AS NO NOT MET)
  10  IF(NC .GE. NO) THEN
        WRITE(6,*) '$GG EVAL NC,SYCSG=',NC,(SYCSG(II),II=1,NYCSG)
        GO TO 60
      ENDIF
      IF(NYCSG.EQ.1) THEN
        CALL GETVYC(SYCS(1),LG,YCSB,NYE,NYB,NYEG,VYC)
        WRITE(6,*) '=== VYC=',(VYC(I),I=1,NYE)
```

```
      ENDIF
      CALL EXPAND(NYCSG,SYCSG,VYC,NYE,IGX,LGX,MGX,LG,MG,RC)
      IF(IGX .EQ. 0) GO TO 20
      NC=NCEVAL(AIMIN,IGX,LGX,MGX,NYCSG,SYCSG,RETTBL,RETINV,NRET,MXX,
     *  LGRP,MGRP,NMS,AGGPCT,IGR,LG)
      GO TO 10
   20 NCTOT=NC
      WRITE(6,*) '$$G EVAL NC,SYCSG=',NCTOT,(SYCSG(II),II=1,NYCSG)
C
C --- EXPAND TO LARGE MOS GROUP
      WRITE(6,*) ' '
      WRITE(6,*) '=== EXPANDING BY LARGE GROUP: ',LGX
      NC=NCEVAL(AIMIN,IGX,LGX,MGX,NYCSL,SYCSL,RETTBL,RETINV,NRET,MXX,
     *  LGRP,MGRP,NMS,AGGPCT,IGR,LG)
   30 IF((NCTOT+NC) .GE. NO) THEN
        WRITE(6,*)'$LL EVAL NC,SYCSL=',(NCTOT+NC),(SYCSL(II),II=1,NYCSL)
        GO TO 60
      ENDIF
      IF(NYCSL.EQ.1) CALL GETVYC(SYCS(1),LG,YCSB,NYE,NYB,NYEG,VYC)
      CALL EXPAND(NYCSL,SYCSL,VYC,NYE,IGX,LGX,MGX,LG,MG,RC)
      IF(LGX .EQ. 0) GO TO 40
      NC=NCEVAL(AIMIN,IGX,LGX,MGX,NYCSL,SYCSL,RETTBL,RETINV,NRET,MXX,
     *  LGRP,MGRP,NMS,AGGPCT,IGR,LG)
      GO TO 30
   40 NCTOT=NCTOT+NC
      WRITE(6,*) '$$L EVAL NC,SYCSL=',NCTOT,(SYCSL(II),II=1,NYCSL)
C
C --- EXPAND TO MAJOR MOS GROUP
      WRITE(6,*) ' '
      WRITE(6,*) '=== EXPANDING BY MAJOR GROUP: ',MGX
      NC=NCEVAL(AIMIN,IGX,LGX,MGX,NYCSM,SYCSM,RETTBL,RETINV,NRET,MXX,
     *  LGRP,MGRP,NMS,AGGPCT,IGR,LG)
   50 IF((NCTOT+NC) .GE. NO  .OR.  RC .NE. 0) THEN
        WRITE(6,*)'$MM EVAL NC,SYCSM=',(NC+NCTOT),(SYCSM(II),II=1,NYCSM)
        GO TO 60
      ENDIF
      IF(NYCSM.EQ.1) CALL GETVYC(SYCS(1),LG,YCSB,NYE,NYB,NYEG,VYC)
      CALL EXPAND(NYCSM,SYCSM,VYC,NYE,IGX,LGX,MGX,LG,MG,RC)
      NC=NCEVAL(AIMIN,IGX,LGX,MGX,NYCSM,SYCSM,RETTBL,RETINV,NRET,MXX,
     *  LGRP,MGRP,NMS,AGGPCT,IGR,LG)
      GO TO 50
C
C --- EXPANSION FINISHED
   60 IF(RC .NE. 0) THEN
        WRITE(5,*)'*** REQUIRED NO MAY NOT BE MET: NO,NC=',NO,(NC+NCTOT)
      ENDIF
C --- ALLOW USER TO CHANGE EXPANSION LEVEL
      WRITE(5,*) 'ESTIMATED NUMBER OF CELLS =',NC+NCTOT
   70 WRITE(5,*)
      WRITE(5,*) 'ENTER 1 TO CALL READER, 0 TO CHANGE EXPANSION'
      READ(5,*) NPICK1
      IF(NPICK1 .EQ. 1) THEN
        GO TO 80
      ELSE
        WRITE(5,*) 'AGGPCT IS CURRENTLY =', AGGPCT
        WRITE(5,*) 'ENTER NEW VALUE FOR AGGPCT'
```

```
        READ(5,*) AGGPCT
        GO TO 5
      ENDIF
   80 WRITE(5,*) 'CALLING READER'
C
C --- USER ELECTS TO READ THE DATA BASE - DETERMINE MOS EXPANSION LEVEL
      CALL GETMOS(SMOS,NMOS,MGX,LGX,MG,LG,IGR,MOSGR,LGRP,MGRP,
     * NMS,NG,NLG)
C --- READ THE DATA BASE AND CREATE THE CELLS
      CALL READER(DATA,INV,Y,MXX,NMOS,NYCSG,NYCSL,NYCSM,NSC,NCSR,NYR,
     * SMOS,SYCSG,SYCSL,SYCSM,SGRD,SVCMP,SCSRC,NRC,PTRTBL,LGX,MGX,IGR,
     * LG,MGRP,LGRP,MOSGR,NMS,NG,NLG,ICYCLE,NPT,PTBL,ISFLAG,SINV,SY)
C --- PERFORM CELL AGGREGATION TO MEET INVENTORY THRESHOLD
      CALL AGGREG(INV,Y,MXX,NYR,SMOS,SYCSG,
     *     NRC, NRCOLD,PTRTBL,INDX,AVINV,AIMIN,MKG)
C --- ALLOW USER TO CHANGE EXPANSION LEVEL
      WRITE(5,*) 'NUMBER OF CELLS =',NRC
   90 WRITE(5,*)
      WRITE(5,*) 'ENTER 1 TO CONTINUE, 0 TO CHANGE EXPANSION'
      READ(5,*) NPICK2
      IF(NPICK2 .EQ. 1) THEN
        GO TO 100
      ELSE
        WRITE(5,*) 'AGGPCT IS CURRENTLY =', AGGPCT
        WRITE(5,*) 'ENTER NEW VALUE FOR AGGPCT'
        READ(5,*) AGGPCT
        ICYCLE=ICYCLE+1
        GO TO 5
      ENDIF
C
C --- USER ELECTS TO CONDUCT ESTIMATION
  100 CONTINUE
      WRITE(11,201)'EXPANSION INFORMATION: '
      WRITE(11,203)'ACTUAL NO. OF CELLS USED= ',NRC
      WRITE(11,202)'MOS GROUP #',IGR,' YCS''S USED=',
     *             (SYCSG(I),I=1,NYCSG)
      IF(LGX .GT. 0) THEN
        WRITE(11,204)'LARGE MOS GROUP #',LG,' YCS''S USED=',
     *                (SYCSL(I),I=1,NYCSL)
      ELSE IF(MGX .GT. 0) THEN
        WRITE(11,204)'LARGE MOS GROUP #',LG,' YCS''S USED=',
     *                (SYCSL(I),I=1,NYCSL)
        WRITE(11,204)'MAJOR MOS GROUP #',MG,' YCS''S USED=',
     *                (SYCSM(I),I=1,NYCSM)
      ENDIF
C --- PERFORM ALL BUT VECTOR ESTIMATION METHODS IN MC87BZ
      CALL MC87BZ(INV,Y,NRC,NYR,XTB,VXTB,XEB,A,MXX,MXY)
C --- VECTOR METHOD--BREAK CELLS INTO VECTOR, CONDUCT ESTIMATION
      IF(ISFLAG .GT. 0) THEN
        CALL BKDOWN(PTBL,NPT,PTRTBL,NRCOLD,INDX,MKG,MXX,MXY,
     *       SINV,SY,INV,Y,BKTBL,NBK)
        CALL MC87V(INV,Y,MXX,NYR,NRC,XTBJI,DELTA,X,XVYR,VYRINV,VYRY,
     *       BSTAR,S,GAMMA,XBBJ,EVAL,MXP,MXK,BKTBL,NBK,NSC,NCSR,ISFLAG)
      ENDIF
C
  201 FORMAT(/1X,A)
```

```
 202  FORMAT(1X,A,I2,A/1X,18(I3))
 203  FORMAT(1X,A,I2)
 204  FORMAT(1X,A,I1,A/1X,18(I3))
      END
C
**************************************************************************
C
      SUBROUTINE EXPAND(NYCSX,SYCSX,VYC,NYE,IGX,LGX,MGX,LG,MG,RC)
C --- EXPAND YCS IF FEASIBLE, ELSE EXPAND MOS TO LG/MG
      INTEGER SYCSX(31), NYCSX
      INTEGER*2 VYC(NYE)
C --- FIND POSITION OF ORIGINALLY REQUESTED SYCS(1)
      IY=0
      DO 10 I=1,NYE
       IF(SYCSX(1) .EQ. VYC(I)) IY=I
  10  CONTINUE
      IF(IY.EQ.0) GO TO 30
C --- FIND NEAREST NON-ZERO YCS TO USE FOR EXPANSION
      DO 20 I=1,NYE
       J=IY-I
       IF(J.GE.1) THEN
            IF(VYC(J).GT.0) GO TO 50
       ENDIF
       J=IY+I
       IF(J.LE.NYE) THEN
            IF(VYC(J).GT.0) GO TO 50
       ENDIF
  20  CONTINUE
  30  CONTINUE
C --- NO MORE YCS EXPANSION POSSIBLE. SEE IF MOS EXP. FEASIBLE
      IF(IGX.GT.0) THEN
C ---         EXPAND FROM GROUPS TO LARGE GROUP LGX
              IGX=0
              LGX=LG
      ELSE IF(LGX.GT.0) THEN
C ---         EXPAND FROM LARGE GROUP LGX TO MAJOR GROUP MGX
              LGX=0
              MGX=MG
      ELSE
              RC=1
      ENDIF
      RETURN
C
C --- EXPAND WITH YCS IN POSITION J & CLEAR VYC(J)
   50 CONTINUE
      NYCSX=NYCSX+1
      SYCSX(NYCSX)=VYC(J)
      VYC(J)=0
      END
C
**************************************************************************
C
      FUNCTION NCEVAL(AIMIN,IGX,LGX,MGX,NYCSX,SYCSX,RETTBL,RETINV,
     *     NRET,MXX,LGRP,MGRP,NMS,AGGPCT,IGR,LG)
C --- COMPUTE ESTIMATED NO. CELLS TO BE OBTAINED WITH CURRENT EXPANSION
      INTEGER SYCSX(31),NYCSX
```

```
      INTEGER*2 LGRP(14),MGRP(6)
      INTEGER*2 RETTBL(MXX, 3)
      REAL RETINV(MXX)
      NCEVAL=0
      IF(IGX.EQ.0 .AND. LGX.EQ.0 .AND. MGX.EQ.0) RETURN
      TAINV=0.0
      DO 100 I=1,NRET
C ---  SCREEN ON YCS
       DO 10 J=1,NYCSX
        IF(RETTBL(I,2) .EQ. SYCSX(J)) GO TO 15
   10  CONTINUE
       GO TO 100
C ---  SCREEN ON MOS BY GROUP, L.GRP OR MG DEPENDING ON IGX,LGX,MGX
   15  CONTINUE
       MOS=RETTBL(I,1)
       IGP=RETTBL(I,3)
       LGP=LGRP(IGP)
       IF(MGX .GT. 0) THEN
          IF(MGRP(LGP) .EQ. MGX) THEN
             IF(LGP .NE. LG) GO TO 80
          ENDIF
       ELSE IF(LGX .GT. 0) THEN
          IF(LGP .EQ. LGX) THEN
             IF(IGP .NE. IGR) GO TO 80
          ENDIF
       ELSE
          IF(IGP .EQ. IGX) GO TO 80
       ENDIF
       GO TO 100
   80  CONTINUE
C ---  ACCEPTED
       IF(RETINV(I) .GE. AIMIN) THEN
              NCEVAL=NCEVAL+1
       ELSE
              TAINV=TAINV+RETINV(I)
       ENDIF
  100 CONTINUE
C --- FINAL ESTIMATE IS NCEVAL
      IF(AIMIN.GT.0) NCEVAL=NCEVAL + AGGPCT*TAINV/AIMIN
      END
C
**************************************************************************
C
      SUBROUTINE GETVYC(SYCS, LG, YCSB,NYE,NYB,NYEG, VYC)
      INTEGER*2 YCSB(NYE,NYB,NYEG), VYC(NYE), LGEX(6)
      INTEGER SYCS
      DATA LGEX/4,4,1,2,4,3/
C --- L INDICATES LAST DIMENSION IN YCS EXPANSION TABLE
      L=LGEX(LG)
C --- FIND TO WHICH YCS BLOCK SYCS BELONGS AND MAKE COPY IN VYC
      DO 10 J=1,NYB
       DO 20 I=1,NYE
        IF(SYCS .EQ. YCSB(I,J,L)) THEN
               DO 30 K=1,NYE
                VYC(K)=YCSB(K,J,L)
  30           CONTINUE
```

```
                RETURN
        ENDIF
  20   CONTINUE
  10  CONTINUE
      WRITE(6,*) '******** YCS NOT FOUND IN YCSB TABLE YCS=',SYCS
      END
C
************************************************************************
C
      SUBROUTINE READET(RETTBL,RETINV,MXX,NRET, SGRD, NSC,SVCMP,
     *            NCSR,SCSRC, MG,LGRP,MGRP, MOSGR,NMS)
C --- READ TABLE WITH ALL EXISTING COMBINATIONS FOR SELECTION CRITERIA
C --- ACCEPT RECS WITH MATCHING PG,MG,CS,SVC.  ACCEPT ALL YCS
      INTEGER SVCMP(5), NSC, SVC
      INTEGER SCSRC(16),NCSR, CS
      INTEGER SGRD, PG
      INTEGER MOS,YCS
      INTEGER*2 MOSGR(2,NMS), MGRP(*),LGRP(*)
      INTEGER*2 RETTBL(MXX, 3)
      REAL RETINV(MXX), AI
      NRET=0
      DO 10 I=1,999999
       READ(10+SGRD,100,END=999) PG,MOS,YCS,SVC,CS, NRECS,AI
       IF(PG .NE. SGRD) GO TO 10
       IGR=IGFIND(MOS, MOSGR,NMS)
       LG=LGRP(IGR)
       IF(MGRP(LG) .NE. MG) GO TO 10
       DO 20 J=1,NSC
        IF(SVC .EQ. SVCMP(J)) GO TO 21
   20  CONTINUE
       GO TO 10
   21  CONTINUE
       DO 30 J=1,NCSR
         IF(CS .EQ. SCSRC(J)) THEN
           CALL ACCEPT(MOS,YCS,IGR,RETTBL,MXX,NRET,RETINV,AI)
           GO TO 10
         ENDIF
   30  CONTINUE
C
   10 CONTINUE
  999 CONTINUE
      IF(NRET .GT. MXX) THEN
        WRITE(6,*) '**** ERROR - TOO MANY RECORDS IN RETTBL'
        STOP
      ENDIF
  100 FORMAT(I2,I4,I3,I2,I3,I4,F7.2)
      END
C
************************************************************************
C
      SUBROUTINE ACCEPT(MOS,YCS,IGR, RETTBL,MXX,NRET,RETINV,AI)
C --- ACCEPT ENTRY.  ACCUMULATE IF ALREADY SAME COMBINATION IS PRESENT
      INTEGER MOS,YCS
      INTEGER*2 RETTBL(MXX, 3)
      REAL RETINV(MXX), AI
      DO 10 I=1,NRET
```

```
       IF(MOS.EQ.RETTBL(I,1) .AND. YCS.EQ.RETTBL(I,2) ) THEN
                        RETINV(I)=RETINV(I) + AI
                        RETURN
       ENDIF
   10 CONTINUE
C --- NEW COMBINATION
      NRET=NRET+1
      RETTBL(NRET,1)=MOS
      RETTBL(NRET,2)=YCS
      RETTBL(NRET,3)=IGR
      RETINV(NRET)=AI
      END
C
**************************************************************************
C
      SUBROUTINE GETPAR(AIMIN,NO,NMOS,SMOS,NYCS,SYCS,SGRD,
     *               NSC,SVCMP, NCSR,SCSRC, IGR,MOSGR,NMS, ISFLAG)
C --- GET SELECTION CRITERIA FROM USER AND VALIDATE
      INTEGER SYCS(31), NYCS
      INTEGER SMOS(20), NMOS
      INTEGER SVCMP(5), NSC
      INTEGER SCSRC(16),NCSR
      INTEGER SGRD
      INTEGER*2 MOSGR(2,NMS)
      WRITE(5,*) ' ENTER THRESHOLD MIN. FOR AVERAGE INVENTORY'
      READ(5,*) AIMIN
      WRITE(5,*) ' ENTER THRESHOLD MIN. FOR NUMBER OF CELLS'
      READ(5,*) NO
      WRITE(5,*) ' THRESHOLDS TO USE AIMIN, NO=',AIMIN,NO
C
      WRITE(5,*) ' ENTER MOS (ONLY 1 ACCEPTED)'
      NMOS=1
      READ(5,*) SMOS(1)
      WRITE(6,*) ' MOS SELECTED: ', SMOS(1)
      IGR=IGFIND(SMOS(1), MOSGR,NMS)
      WRITE(6,*) ' GROUP TO USE: ', IGR
      IF(IGR.EQ.0) THEN
              WRITE(5,*) '**** ERROR - INVALID MOS SELECTED: ',SMOS(1)
              STOP
      ENDIF
C
      WRITE(5,*) ' ENTER YCS (ONLY 1 ACCEPTED)'
      NYCS=1
      READ(5,*) SYCS(1)
      WRITE(6,*) ' YCS SELECTED: ', SYCS(1)
C
      WRITE(5,*) ' ENTER GRADE'
      READ(5,*) SGRD
      WRITE(6,*) ' GRADE SELECTED', SGRD
C
      WRITE(5,*) ' ENTER NO. OF SVC. COMPS & ARRAY (1-3, 4=1+2, 5=ALL)'
      READ(5,*) NSC, (SVCMP(I), I=1,NSC)
C --- EXPAND 4 TO 1,2 AND 5 TO 1,2,3
      DO 10 I=1,NSC
       IF(SVCMP(I).EQ.4 .OR. SVCMP(I).EQ.5) THEN
          NSC=SVCMP(I)-2
```

```
          DO 15 J=1,NSC                                                 MC804470
           SVCMP(J)=J                                                   MC804480
   15     CONTINUE                                                      MC804490
          GO TO 11                                                      MC804500
       ENDIF                                                            MC804510
   10 CONTINUE                                                          MC804520
   11 CONTINUE                                                          MC804530
      WRITE(6,*) ' SERVICE COMPONENTS SELECTED', (SVCMP(I), I=1,NSC)    MC804540
C                                                                       MC804550
      WRITE(5,*) ' ENTER NO. COMM. SOURCES AND ARRAY (1-15, 16=ALL)'    MC804560
      READ(5,*) NCSR, (SCSRC(I), I=1,NCSR)                              MC804570
C --- IF 16 IS SELECTED THEN EXPAND ARRAY TO COVER ALL 1-15             MC804580
      DO 20 I=1,NCSR                                                    MC804590
       IF(SCSRC(I) .EQ. 16) THEN                                        MC804600
          NCSR=15                                                       MC804610
          DO 25 J=1,NCSR                                                MC804620
           SCSRC(J)=J                                                   MC804630
   25     CONTINUE                                                      MC804640
          GO TO 26                                                      MC804650
       ENDIF                                                            MC804660
   20 CONTINUE                                                          MC804670
   26 CONTINUE                                                          MC804680
      WRITE(5,*) ' COMM. SOURCES SELECTED:', (SCSRC(I), I=1,NCSR)       MC804690
C                                                                       MC804700
C --- FLAG TO DETERMINE WHICH OF SVC OR CS WILL BE USED AS 3RD DIMENSIONMC804710
      WRITE(5,*) 'SELECT 3RD DIM. TO USE: 0=NONE, 1=SVC, 2=COMM. SOURCE'MC804720
      READ(5,*) ISFLAG                                                  MC804730
C --- WRITE INPUT PARAMETER INFO TO OUTPUT FILE                         MC804740
      WRITE(11,101) 'TEST CASE INPUT PARAMETERS:'                       MC804750
      WRITE(11,102) 'INVENTORY THRESHOLD= ',AIMIN,                      MC804760
     *              'THRESHOLD NO. OF CELLS= ',NO                       MC804770
      WRITE(11,103) 'MOS= ',SMOS(1),'YCS= ',SYCS(1),'GRADE= ',SGRD      MC804780
      WRITE(11,104) 'SERVICE COMPONENTS= ',(SVCMP(I),I=1,NSC)           MC804790
      WRITE(11,104) 'COMM SOURCES= ',(SCSRC(I),I=1,NCSR)                MC804800
      WRITE(6,*)   '3RD DIMENSION= ',ISFLAG                             MC804810
C                                                                       MC804820
 101  FORMAT(1X,A)                                                      MC804830
 102  FORMAT(1X,A,F4.1,7X,A,I2)                                         MC804840
 103  FORMAT(1X,A,I3,2(5X,A,I2))                                        MC804850
 104  FORMAT(1X,A,15(I3))                                               MC804860
      END                                                               MC804870
C                                                                       MC804880
************************************************************************MC804890
C                                                                       MC804900
      SUBROUTINE GETMOS(SMOS,NMOS,MGX,LGX,MG,LG,IGR,MOSGR,LGRP,MGRP,    MC804910
     *  NMS,NG,NLG)                                                     MC804920
C --- BUILD SMOS ARRAY BASED UPON EXPANSION                             MC804930
      INTEGER SMOS(30)                                                  MC804940
      INTEGER*2 MOSGR(2,NMS), LGRP(NG), MGRP(NLG)                       MC804950
      NMOS=0                                                            MC804960
      IF(MGX .GT. 0) THEN                                               MC804970
C ---   HAVE EXPANDED TO MAJOR MOS GROUP                                MC804980
        DO 10  I=1,NMS                                                  MC804990
          IGP=MOSGR(2,I)                                                MC805000
          LGP=LGRP(IGP)                                                 MC805010
          IF(MGRP(LGP) .EQ. MG) THEN                                    MC805020
```

```
            NMOS=NMOS+1
            SMOS(NMOS)=MOSGR(1,I)
          ENDIF
   10   CONTINUE
        RETURN
      ELSE IF(LGX .GT. 0) THEN
C ---   HAVE EXPANDED TO LARGE MOS GROUP
        DO 20  I=1,NMS
          IGP=MOSGR(2,I)
          IF(LGRP(IGP) .EQ. LG) THEN
            NMOS=NMOS+1
            SMOS(NMOS)=MOSGR(1,I)
          ENDIF
   20   CONTINUE
        RETURN
      ELSE
C ---   HAVE EXPANDED TO SMALL MOS GROUP
        DO 30  I=1,NMS
          IF(MOSGR(2,I) .EQ. IGR) THEN
            NMOS=NMOS+1
            SMOS(NMOS)=MOSGR(1,I)
          ENDIF
   30   CONTINUE
        RETURN
      ENDIF
      END
C
************************************************************************
C
      FUNCTION IGFIND(MOS, MOSGR,NMS)
C --- FIND LOCATION OF MATCHING MOS IN GROUP TABLE. RETURN GROUP NO
      INTEGER*2 MOSGR(2,NMS)
      DO 10 I=1,NMS
        IF(MOSGR(1,I) .EQ. MOS) THEN
                                     IGFIND=MOSGR(2,I)
                                     RETURN
        ENDIF
   10 CONTINUE
      IGFIND=0
      END
C
************************************************************************
C
      SUBROUTINE READER(DATA,INV,Y,MXX,NMOS,NYCSG,NYCSL,NYCSM,NSC,NCSR,
     * NYR,SMOS,SYCSG,SYCSL,SYCSM,SGRD,SVCMP,SCSRC,NRC,PTRTBL,LGX,MGX,
     * IGR,LG,MGRP,LGRP,MOSGR,NMS,NG,NLG,ICYCLE,NPT,PTBL,ISFLAG,SINV,SY)
      REAL INV(MXX,NYR),Y(MXX,NYR), SINV(MXX,NYR),SY(MXX,NYR)
      INTEGER*2 PTRTBL(MXX, 2), PTBL(MXX,3)
      INTEGER SYCSG(*), SYCSL(*), SYCSM(*)
      INTEGER SMOS(*), NMOS
      INTEGER SVCMP(*), NSC
      INTEGER SCSRC(*),NCSR
      INTEGER SGRD
      INTEGER TYPE,YCS,PG,MOS,SEX,CS,EDLV,SVC,MOS1,MOS2,RACE
      INTEGER DATA(NYR)
      CHARACTER*7 CITLS
```

```
      INTEGER*2 MOSGR(2,NMS),LGRP(NG),MGRP(NLG)
C
C --- REWIND DATA FILE AND RESET INV,Y IF CYCLING THRU READER
      IF(ICYCLE .GT. 1) THEN
        REWIND 1
        DO 6  I=1,MXX
          DO 5  J=1,NYR
            INV(I,J)=0.0
              Y(I,J)=0.0
           SINV(I,J)=0.0
             SY(I,J)=0.0
  5       CONTINUE
  6     CONTINUE
      ENDIF
C --- READ RECORD AND STORE IN MATRIX
      ICR=0
      NRC=0
      NPT=0
      ICNT=0
      IYNO=0
      IYR=0
C
    1 READ(1,101,END=999) TYPE,YCS,PG,MOS,SEX,CS,EDLV,SVC,MOS1,MOS2,
     *        RACE,CITLS,DATA
      ICR=ICR+1
C --- CHECK IF RECORD MEETS SELECTION CRITERIA.  OTHERWISE REJECT.
C --- COLLECT TYPES 0=INVENTORY, AND 1-5 ALL LOSSES
      IF(TYPE.GT.5) GO TO 999
C
C --- SCREEN FOR GRADE
      IF(PG .NE. SGRD) GO TO 1
C
C --- SCREEN FOR MOS
      IGP=IGFIND(MOS,MOSGR,NMS)
      IF(IGP.EQ.0) GO TO 1
      LGP=LGRP(IGP)
      IF(MGX .GT. 0) THEN
C ---   HAVE EXPANDED TO MAJOR MOS GROUP
        IF(LGP .EQ. LG) THEN
          DO 10  I=1,NYCSL
            IF(YCS .EQ. SYCSL(I)) THEN
              IY=I
              GO TO 60
            ENDIF
  10      CONTINUE
          GO TO 1
        ELSE IF(MGRP(LGP) .EQ. MGX) THEN
          DO 20  I=1,NYCSM
            IF(YCS .EQ. SYCSM(I)) THEN
              IY=I
              GO TO 60
            ENDIF
  20      CONTINUE
          GO TO 1
        ELSE
          GO TO 1
```

```
         ENDIF
       ELSE IF(LGX .GT. 0) THEN
C ---    HAVE EXPANDED TO LARGE MOS GROUP
         IF(IGP .EQ. IGR) THEN
           DO 30  I=1,NYCSG
             IF(YCS .EQ. SYCSG(I)) THEN
               IY=I
               GO TO 60
             ENDIF
   30      CONTINUE
           GO TO 1
         ELSE IF(LGP .EQ. LGX) THEN
           DO 40  I=1,NYCSL
             IF(YCS .EQ. SYCSL(I)) THEN
               IY=I
               GO TO 60
             ENDIF
   40      CONTINUE
           GO TO 1
         ELSE
           GO TO 1
         ENDIF
       ELSE
C ---    HAVE EXPANDED TO SMALL MOS GROUP
         IF(IGP .EQ. IGR) THEN
           DO 50  I=1,NYCSG
             IF(YCS .EQ. SYCSG(I)) THEN
               IY=I
               GO TO 60
             ENDIF
   50      CONTINUE
           GO TO 1
         ELSE
           GO TO 1
         ENDIF
       ENDIF
   60  CONTINUE
C
       DO 70  I=1,NMOS
         IF(MOS .EQ. SMOS(I)) THEN
           IM=I
           GO TO 80
         ENDIF
   70  CONTINUE
       WRITE(6,*) '*** ERROR IN MOS SCREENING ***',MOS
       WRITE(6,*) 'NMOS,SMOS=',NMOS,(SMOS(I),I=1,NMOS)
       GO TO 1
C
C ---  SCREEN FOR SERVICE COMPONENT
   80  CONTINUE
       DO 90 I=1,NSC
        IF(SVC .EQ. SVCMP(I)) THEN
               IS=I
               GO TO 100
        END IF
   90  CONTINUE
```

```
      GO TO 1
C
C --- SCREEN FOR COMMISSIONING SOURCE
 100  CONTINUE
      DO 110 I=1,NCSR
       IF(CS .EQ. SCSRC(I)) THEN
              IR=I
              GO TO 120
       END IF
 110  CONTINUE
      GO TO 1
C
 120  CONTINUE
C
C --- RECORD ACCEPTED - INSTALL IT IN INV,Y,SINV,SY, PTRTBL AND PTBL
      ICNT=ICNT+1
      IF(ISFLAG.EQ.1) THEN
                IW=IS
      ELSE IF(ISFLAG.EQ.2) THEN
                IW=IR
      ELSE
                IW=-99
      ENDIF
      MINV=GINV(PTRTBL, MXX,NRC, IM,IY,-99)
      MV=GINV(PTBL, MXX,NPT,IM,IY,IW)
      IF(TYPE.EQ.0) THEN
          CALL INSINV(PTRTBL,MXX,NYR,NRC,MINV,IM,IY,-99,INV,DATA)
          CALL INSINV(PTBL,  MXX,NYR,NPT,MV,  IM,IY, IW,SINV,DATA)
      ELSE
          CALL INSY(MXX,NYR,MINV,Y,DATA)
          CALL INSY(MXX,NYR,MV,  SY,DATA)
          IYR=IYR+1
          IF(MINV.EQ.0) THEN
            WRITE(6,*) '*** ERROR IN DATA BASE.  LOSS  W/O INV. REC.'
            WRITE(6,122) 'Y**:', MOS,YCS,PG,EDLV,SVC,RACE,
     *                           (DATA(IT), IT=1,NYR)
            IYNO=IYNO+1
          ENDIF
      ENDIF
C
      GO TO 1
C
  999 CONTINUE
      WRITE(6,*) ' '
      WRITE(6,*) 'TOTAL RECORDS READ                   =',ICR
      WRITE(6,*) 'TOTAL INV. MOS/YCS    COMBINATIONS=',NRC
      WRITE(6,*) 'TOTAL INV. MOS/YCS/IW COMBINATIONS=',NPT
      WRITE(6,*) 'TOTAL RECORDS ACCEPTED               =',ICNT
      WRITE(6,*) 'TOTAL LOSS RECORDS ACCEPTED          =',IYR
      WRITE(6,*) 'TOTAL LOSS RECORDS NOT MATCHED       =',IYNO
C --- TERMINATE IF NO DATA COLLECTED
      IF(NRC .EQ. 0) THEN
              WRITE(6,*) '**** NO DATA MEETS SELECTION REQS'
              STOP
      ENDIF
C
```

```
 101  FORMAT(3I2,I3,I1,I2,2I1,2I3,I1,A7, 1X, 10I4)
 121  FORMAT(A8,13I6)
 122  FORMAT(A8,7I6, 5X, 12I6)
 131  FORMAT(I4, 2I6)
 132  FORMAT(I4, 3I6, 10F7.2)
      END
C
*******************************************************************************
C
      FUNCTION GINV(PTBL, MXX,NPT, IM,IY,IW)
C --- FIND LOCATION OF INVENTORY ENTRY FOR MOS,YCS,SVC/CS COMBINATIONS
C --- 3RD DIMENSION CHECKED ONLY IN CASE IW>0
      INTEGER*2 PTBL(MXX, *)
      DO 10 I=1,NPT
        IF(PTBL(I, 1) .EQ. IM  .AND.
     *     PTBL(I, 2) .EQ. IY ) THEN
           IF(IW.LT.0 .OR. (IW.GT.0 .AND. PTBL(I, 3).EQ.IW)) THEN
                                        GINV=I
                                        RETURN
           ENDIF
        ENDIF
   10 CONTINUE
      GINV=0
      END
C
*******************************************************************************
C
      SUBROUTINE INSINV(PT,MXX,NYR,N,K,IM,IY,IW,INV,DATA)
C --- ACCUMM INTO KTH ENTRY.  INSTALL IN POINTER TABLE IF NOT PRESENT
      REAL INV(MXX, NYR)
      INTEGER*2 PT(MXX, *)
      INTEGER DATA(NYR)
      IF(K .EQ. 0) THEN
C ---       ADD NEW ENTRY
            N=N+1
            IF(N .GT. MXX) THEN
              WRITE(6,*) '*** ERROR - TOO MANY INV. COMBINATIONS',N
              STOP
            ENDIF
            K=N
            PT(K, 1)=IM
            PT(K, 2)=IY
            IF(IW.GT.0) PT(K, 3)=IW
      ENDIF
      DO 130 IT=1,NYR
       INV(K,IT)=INV(K,IT) + .25*FLOAT(DATA(IT))
 130  CONTINUE
      END
C
*******************************************************************************
C
      SUBROUTINE INSY(MXX,NYR,K,Y,DATA)
C --- ACCUMM INTO KTH ENTRY FOR LOSS
      REAL Y(MXX, NYR)
      INTEGER DATA(NYR)
      IF(K .EQ. 0) RETURN
```

```
      DO 10 IT=1,NYR
       Y(K,IT)=Y(K,IT) + DATA(IT)
   10 CONTINUE
      END
C
******************************************************************************
C
      SUBROUTINE AGGREG(INV,Y,MXX,NYR,SMOS,SYCSG,
     *                 NRC,NRCOLD,PTRTBL,INDX,AVINV, AIMIN,MKG)
C --- COMP. AVERAGE INV. & SORT
      REAL INV(MXX, NYR), Y(MXX, NYR), AVINV(MXX)
      INTEGER*2 PTRTBL(MXX, 2), INDX(MXX),MKG(MXX)
      INTEGER SYCSG(*), SMOS(*)
      REAL*8 TINV,TY
C
C --- RESET MKG (NECESSARY WHEN CYCLING THRU AGGPCT VALUES)
      DO 10  I=1,MXX
        MKG(I)=0
  10  CONTINUE
      TINV=0
      TY=0
      DO 100 I=1,NRC
C ---  FIX INV. ENTRIES LOWER THAN CORRESP. LOSSES & COMP. AVG INV.
       AI=0
       DO 201 J=1,NYR
        TINV=TINV+INV(I,J)
          TY=  TY+  Y(I,J)
        IF(INV(I,J).LT.Y(I,J)) INV(I,J)=Y(I,J)
        AI=AI+INV(I,J)
  201  CONTINUE
       AVINV(I)=AI/NYR
       INDX(I)=I
  100 CONTINUE
      WRITE(6,*) '===== TOTAL INV,Y=',TINV,TY
C
C --- SORT ASCENDING BY  AVG INVENTORY
      CALL SORT2(AVINV,INDX,NRC)
C
      NS1=0
C --- DISPLAY TABLE IN SORT SEQUENCE
      CALL DSPTBL(INV,Y,AVINV,PTRTBL,INDX,AIMIN,NRC,MKG,MXX,NYR,
     *                  SYCSG,SMOS )
C
      DO 200 K=NRC,1,-1
       IF(AVINV(K) .GE. AIMIN) THEN
C ---           MARK AS MEMBER OF SET S0
                MKG(K)=32767
       ELSE
C ---           INITIAL COUNT OF MEMBERS OF SET S1
                NS1=K
                GO TO 202
       ENDIF
  200 CONTINUE
  202 CONTINUE
C --- DO AGGREGATIONS WITHIN SET S1 UNTIL NO MORE POSSIBLE (KF GE 0)
```

```
      KF=-1
C --- DO WHILE KF<0
  300 IF(KF.GE.0) GO TO 310
       CALL AGG1(AVINV,INDX,MKG,NS1,INV,Y,MXX,NYR,AIMIN,KF)
       GO TO 300
  310 CONTINUE
C --- DISPLAY TABLE AFTER 1ST AGGREGATION
      CALL DSPTBL(INV,Y,AVINV,PTRTBL,INDX,AIMIN,NRC,MKG,MXX,NYR,
     *                  SYCSG,SMOS )
      IF(NS1.EQ.NRC) THEN
           WRITE(6,*) '***** SET S0 EMPTY. NO CELLS ABOVE THRESHOLD'
           STOP
      ENDIF
C --- DO AGGREGATIONS FROM SET S1 INTO SET S0 UNTIL NO MORE POSSIBLE
      KF=1
C --- DO WHILE KF>0
  320 IF(KF.LE.0) GO TO 330
         CALL AGG2(AVINV,INDX,MKG,NS1,NRC,INV,Y,MXX,NYR, KF)
         GO TO 320
  330 CONTINUE
C --- DISPLAY TABLE AFTER 2ND AGGREGATION
      CALL DSPTBL(INV,Y,AVINV,PTRTBL,INDX,AIMIN,NRC,MKG,MXX,NYR,
     *                  SYCSG,SMOS )
C --- MOVE VALUES GE AIMIN TO BEGINNING OF ARRAYS
      CALL CMPRS(INV,Y,MXX,NYR,NRC,NRCOLD,AIMIN,AVINV)
C --- DISPLAY TABLE AFTER MOVING VALUES.
      DO 400 K=1,NRC
       WRITE(6,122)K,AVINV(K), (INV(K,J),J=1,NYR)
       WRITE(6,123)            (  Y(K,J),J=1,NYR)
  400 CONTINUE
  122 FORMAT(/I5,14X,F8.3, 6X,    10F7.2)
  123 FORMAT(    33X, 10F7.2)
      END
C ****************
      SUBROUTINE AGG1(AVINV,INDX,MKG,NS1,INV,Y,MXX,NYR,AIMIN,KF)
C --- DO ONE PASS OF AGGREGATION
      REAL INV(MXX, NYR), Y(MXX, NYR), AVINV(MXX)
      INTEGER*2 INDX(MXX),MKG(MXX)
      KF=0
      CI=0
      DO 10 I=NS1,1,-1
       IF(MKG(I).EQ.0) THEN
         IF(KF.EQ.0) THEN
C ---       THIS WILL BE THE COLLECTING CELL
            KF=I
            CI=AVINV(I)
         ELSE
            IF(CI+AVINV(I).LT.AIMIN) THEN
C ---           ACCUM. WITH CELL KF TEMPORARILY. SET TEMP. POINTER -KF
                CI=CI+AVINV(I)
                MKG(I)=-KF
            ELSE
C ---           FIND SMALLEST CELL TO ADD
                CALL AGG1A(AVINV,MKG,I,CI,AIMIN,KF,MXX)
            ENDIF
```

```
            IF(CI.GE.AIMIN) THEN                                               MC808930
C ---           MAKE THIS AGGREGATION PERMANENT AND EXIT                       MC808940
                AVINV(KF)=CI                                                   MC808950
                CALL AGG1B(INDX,MKG,KF,INV,Y,NYR,MXX)                          MC808960
                NS1=NS1-1                                                      MC808970
                MKG(KF)=32767                                                  MC808980
                KF=-1                                                          MC808990
                RETURN                                                         MC809000
            ENDIF                                                              MC809010
         ENDIF                                                                 MC809020
       ENDIF                                                                   MC809030
   10 CONTINUE                                                                 MC809040
C                                                                              MC809050
      IF(KF.EQ.0) RETURN                                                       MC809060
C --- CLEAR TEMPORARY POINTERS LEFT.  THIS WAS AN UNSUCCESSFUL AGGREG.         MC809070
      DO 20 I=1,NS1                                                            MC809080
       IF(MKG(I).LT.0) MKG(I)=0                                                MC809090
   20 CONTINUE                                                                 MC809100
      END                                                                      MC809110
C ****************                                                             MC809120
      SUBROUTINE AGG1A(AVINV,MKG,ILAST,CI,AIMIN,KF,MXX)                        MC809130
C --- FIND SMALLEST CELL TO ADD AND SET TEMPORARY POINTER                      MC809140
      REAL AVINV(MXX)                                                          MC809150
      INTEGER*2 MKG(MXX)                                                       MC809160
      DO 10 I=1,ILAST                                                          MC809170
       IF(MKG(I).EQ.0) THEN                                                    MC809180
        IF(CI+AVINV(I) .GE. AIMIN) THEN                                        MC809190
          CI=CI+AVINV(I)                                                       MC809200
          MKG(I)=-KF                                                           MC809210
          RETURN                                                               MC809220
        ENDIF                                                                  MC809230
       ENDIF                                                                   MC809240
   10 CONTINUE                                                                 MC809250
      WRITE(6,*) '*** ERROR IN AGG1A. NO VALUE FOUND ***'                      MC809260
      STOP                                                                     MC809270
      END                                                                      MC809280
C ****************                                                             MC809290
      SUBROUTINE AGG1B(INDX,MKG,KF,INV,Y,NYR,MXX)                              MC809300
C --- MAKE AGGREGATION PERMANENT                                               MC809310
      REAL INV(MXX, NYR), Y(MXX, NYR)                                          MC809320
      INTEGER*2 INDX(MXX),MKG(MXX)                                             MC809330
      K=INDX(KF)                                                               MC809340
      DO 10 I=1,KF-1                                                           MC809350
       IF(MKG(I) .LT. 0) THEN                                                  MC809360
          IF(MKG(I).NE.-KF) STOP 777                                           MC809370
          MKG(I)=KF                                                            MC809380
          L=INDX(I)                                                            MC809390
          DO 20 J=1,NYR                                                        MC809400
           INV(K,J)=INV(K,J)+INV(L,J)                                          MC809410
             Y(K,J)=  Y(K,J)+  Y(L,J)                                          MC809420
  20      CONTINUE                                                             MC809430
       ENDIF                                                                   MC809440
  10  CONTINUE                                                                 MC809450
      END                                                                      MC809460
C ****************                                                             MC809470
```

```
      SUBROUTINE AGG2(AVINV,INDX,MKG,NS1,NRC,INV,Y,MXX,NYR, KF)
C --- DO ONE PASS OF AGGREGATION FROM SET S1 TO SET SO
C --- ON EACH PASS ONE ELEMENT OF S1 IS TAKEN & ADDED TO SMALLEST OF SO
      REAL INV(MXX, NYR), Y(MXX, NYR), AVINV(MXX)
      INTEGER*2 INDX(MXX),MKG(MXX)
      KF=0
C --- FIND ELEMENT OF S1  (ONLY THOSE WITH POINTER MKG(I)=0)
      DO 10 I=1,NS1
       IF(MKG(I).EQ.0) THEN
             KF=I
             GO TO 12
       ENDIF
   10 CONTINUE
   12 CONTINUE
C --- IF KF STILL 0 THEN NO MORE ELEMENTS IN S1 LEFT
      IF(KF.EQ.0) RETURN
C
C --- FIND SMALLEST ELEMENT OF SO AND ADD TO IT. ONLY WITH MKG(I)=32767
      ISM=NRC
      SMALL=AVINV(ISM)
      DO 20 I=1, NRC
       IF(MKG(I).EQ.32767) THEN
        IF(AVINV(I).LT.SMALL) THEN
           ISM=I
           SMALL=AVINV(I)
        ENDIF
       ENDIF
   20 CONTINUE
C --- JOIN ELEMENT KF TO ELEMENT ISM
      AVINV(ISM)=AVINV(ISM) + AVINV(KF)
      MKG(KF)=ISM
      L=INDX(KF)
      K=INDX(ISM)
      DO 30 J=1,NYR
       INV(K,J)=INV(K,J)+INV(L,J)
         Y(K,J)=  Y(K,J)+  Y(L,J)
  30  CONTINUE
      END
C ****************
      SUBROUTINE CMPRS(INV,Y,MXX,NYR,NRC,NRCOLD,AIMIN,AVINV)
      REAL INV(MXX, NYR), Y(MXX, NYR), AVINV(MXX)
C --- COMPRESS INV,Y IN PLACE.  MOVE ALL ROWS GE AIMIN TO TOP
      NRCOLD=NRC
      NRC=0
      DO 10 I=1,NRCOLD
       AI=CAINV(INV,I,MXX,NYR)
       IF(AI .GE. AIMIN) THEN
C ---       TRANSFER ACTIVE CELL I ---> NRC
           NRC=NRC+1
           AVINV(NRC)=AI
           DO 20 J=1,NYR
            INV(NRC,J)=INV(I,J)
              Y(NRC,J)=  Y(I,J)
  20       CONTINUE
       ENDIF
  10  CONTINUE
```

```
      END
C *****************
      FUNCTION CAINV(INV,I,MXX,NYR)
      REAL INV(MXX, NYR)
C --- COMPUTE AVERAGE INVENTORY FOR ROW I
       CAINV=0
       DO 10 J=1,NYR
        CAINV=CAINV+INV(I,J)
   10  CONTINUE
       CAINV=CAINV/NYR
      END
C *****************
      SUBROUTINE DSPTBL(INV,Y,AVINV,PTRTBL,INDX,AIMIN,NRC,MKG,MXX,NYR,
     *                  SYCSG,SMOS)
C --- DISPLAY TABLE IN SORT SEQUENCE
      REAL INV(MXX, NYR), Y(MXX, NYR), AVINV(MXX)
      INTEGER*2 PTRTBL(MXX, 2), INDX(MXX),MKG(MXX)
      INTEGER SYCSG(*)
      INTEGER SMOS(*)
      INTEGER IATT(2)
      CHARACTER*1 STI
      WRITE(6,121)
      WRITE(6,*) 'INV. THRESHOLD MIN. VALUE=',AIMIN
C
      WRITE(6,*) '   I  INDX   AVG        MKG        INVENTORY/LOSSES'
      DO 200 K=1,NRC
       STI=' '
       I=INDX(K)
       AI=AVINV(K)
       IF(AI .LT. AIMIN) STI='$'
       IATT(1)=SMOS(PTRTBL(I,1))
       IATT(2)=SYCSG(PTRTBL(I,2))
       WRITE(6,122)K,I,AI,MKG(K),STI,(INV(I,J),J=1,NYR),(IATT(J),J=1,2),
     *             PTRTBL(I,1),PTRTBL(I,2)
       WRITE(6,123)                     (  Y(I,J),J=1,NYR)
  200 CONTINUE
C
  121 FORMAT(///)
  122 FORMAT(/2I5,F8.3,I9,1X,A2, 10F7.2, 5X, 6I5)
  123 FORMAT( 30X, 10F7.2)
      END
C *****************
      SUBROUTINE SORT2(Y,INDX, N)
C --- INPLACE SORT USING SHELL ALGORITHM ********
C --- SORTS ON Y AND DOES SAME REORDERING ON INDEXES INDX
      REAL Y(N),TEMP
      INTEGER GAP
      INTEGER*2 INDX(N), ITEMP
      LOGICAL EXCH
C
      GAP=(N/2)
 5    IF (.NOT.(GAP.NE.0)) GO TO 500
10       CONTINUE
            EXCH=.TRUE.
            K=N-GAP
            DO 200 I=1,K
```

```
            KK=I+GAP
            IF(.NOT.(Y(I).GT.Y(KK))) GO TO 100
                  TEMP=Y(I)
                  Y(I)=Y(KK)
                  Y(KK)=TEMP
                  ITEMP=INDX(I)
                  INDX(I)=INDX(KK)
                  INDX(KK)=ITEMP
                  EXCH=.FALSE.
 100        CONTINUE
 200        CONTINUE
         IF (.NOT.(EXCH)) GO TO 10
         GAP=(GAP/2)
      GO TO 5
 500  CONTINUE
      RETURN
      END
C
********************************************************************************
C
      SUBROUTINE BKDOWN(PTBL,NPT,PTRTBL,NRC,INDX,MKG,MXX,MXY,
     *    SINV,SY,INV,Y,BKTBL,NBK   )
C --- BREAKDOWN AGGREGATED VALUES BY THE 3RD DIMENSION SVC/CS
      REAL INV(MXX,MXY),Y(MXX,MXY), SINV(MXX,MXY),SY(MXX,MXY)
      INTEGER*2 PTRTBL(MXX, 2), INDX(MXX), MKG(MXX)
      INTEGER*2 PTBL(MXX, 3), BKTBL(MXX,3)
      REAL*8 TINV,TY
      NBK=0
C --- TRAVERSE MKG ARRAY AND BUILD BKTBL
      DO 10 I=1,NRC
       IF(MKG(I).NE.32767) THEN
                         ICELL=MKG(I)
       ELSE
                         ICELL=I
       ENDIF
       IX=INDX(I)
       IM=PTRTBL(IX,1)
       IY=PTRTBL(IX,2)
       CALL BLDBK(ICELL,IM,IY,PTBL,NPT,MXX,BKTBL,NBK)
   10 CONTINUE
C --- DISPLAY BKTBL PRIOR TO SORTING
      WRITE(6,101) (I,(BKTBL(I,J),J=1,3), I=1,NBK)
      CALL SORT3(BKTBL,NBK,MXX)
      WRITE(6,101) (I,(BKTBL(I,J),J=1,3), I=1,NBK)
C --- SUMMARIZE SINV,SY INTO INV,Y FOR MATCHING ENTRIES IN BKTBL
      CALL SUMBK(BKTBL,NBK,MXX,SINV,SY,INV,Y,MXY)
      WRITE(6,102) (I,(INV(I,J),J=1,MXY),(BKTBL(I,J),J=1,2), I=1,NBK)
      WRITE(6,102) (I,(  Y(I,J),J=1,MXY),(BKTBL(I,J),J=1,2), I=1,NBK)
  101 FORMAT(I4, 3I6)
  102 FORMAT(I4, 10F7.2,10X,2I4)
  103 FORMAT(/I5,10F7.2)
  104 FORMAT( 5X,10F7.2)
      END
C ****************
      SUBROUTINE BLDBK(ICELL,IM,IY,PTBL,NPT,MXX,BKTBL,NBK)
```

```
      INTEGER*2 PTBL(MXX, 3), BKTBL(MXX,3)
C --- RECORD ALL ENTRIES IN PTBL WITH MATCHING IM,IY IN BKTBL
      DO 10 I=1,NPT
       IF(PTBL(I,1).EQ.IM .AND. PTBL(I,2).EQ.IY) THEN
C ---          INSTALL WITH CELL ID, IW & POINTER
             NBK=NBK+1
             BKTBL(NBK,1)=ICELL
             BKTBL(NBK,2)=PTBL(I,3)
             BKTBL(NBK,3)=I
       ENDIF
  10  CONTINUE
      END
C ***************
      SUBROUTINE SORT3(T,N,MXX)
C --- INPLACE SORT USING SHELL ALGORITHM ********
C --- SORTS ON 1ST 2 COLS. OF T & DOES SAME REORDERING ON 3RD COLUMN
      INTEGER*2 T(MXX,3), ITEMP
      INTEGER GAP
      LOGICAL EXCH
C
      GAP=(N/2)
  5   IF (GAP.EQ.0) GO TO 500
 10      CONTINUE
            EXCH=.FALSE.
            K=N-GAP
            DO 200 I=1,K
            KK=I+GAP
            IF(T(I,1).GT.T(KK,1)  .OR.
     *               (T(I,1).EQ.T(KK,1) .AND. T(I,2).GT.T(KK,2)) ) THEN
                  IT1=T(I,1)
                  IT2=T(I,2)
                  IT3=T(I,3)
                  T(I,1)=T(KK,1)
                  T(I,2)=T(KK,2)
                  T(I,3)=T(KK,3)
                  T(KK,1)=IT1
                  T(KK,2)=IT2
                  T(KK,3)=IT3
                  EXCH=.TRUE.
            ENDIF
 200        CONTINUE
         IF (EXCH) GO TO 10
         GAP=(GAP/2)
      GO TO 5
 500  CONTINUE
      RETURN
      END
C ****************
      SUBROUTINE SUMBK(BKTBL,NBK,MXX,SINV,SY,INV,Y,MXY)
C --- CREATE AGGREGATED ARRAYS INV,Y FROM CELL & 3RD DIM. INFO. IN BKTBL
      REAL INV(MXX,MXY),Y(MXX,MXY), SINV(MXX,MXY),SY(MXX,MXY)
      INTEGER*2 BKTBL(MXX,3)
      REAL*8 TINV,TY
      IP=0
      I1=-1
      I2=-1
```

```
      TINV=0
      TY=0
      DO 10 I=1,NBK
       IF(BKTBL(I,1).NE.I1 .OR. BKTBL(I,2).NE.I2) THEN
C ---              CHANGE OF CELL,IW IDENTIFIERS
                  IP=IP+1
                  I1=BKTBL(I,1)
                  I2=BKTBL(I,2)
                  DO 15 J=1,MXY
                   INV(IP,J)=0
                     Y(IP,J)=0
   15             CONTINUE
                  BKTBL(IP,1)=I1
                  BKTBL(IP,2)=I2
       ENDIF
C ---  ACCUMULATE
       I3=BKTBL(I,3)
       DO 20 J=1,MXY
        INV(IP,J)=INV(IP,J)+SINV(I3,J)
          Y(IP,J)=  Y(IP,J)+  SY(I3,J)
        TINV=TINV+SINV(I3,J)
          TY=  TY+  SY(I3,J)
   20  CONTINUE
   10 CONTINUE
      WRITE(6,*) '==== TOTAL INV,Y AFTER BREAKDOWN=',TINV,TY
C
      NBK=IP
C --- FIX INV. ENTRIES LOWER THAN CORRESP. LOSSES
      DO 40  I=1,NBK
        DO 30  J=1,MXY
          IF(INV(I,J).LT.Y(I,J)) INV(I,J)=Y(I,J)
   30   CONTINUE
   40 CONTINUE
      END
```

C. ESTIMATION SUBROUTINES

```
      SUBROUTINE MC87BZ(INV,Y,NRC,NYR,XTB,VXTB,XEB,A,MXX,MXY)
C --- CONDUCTS FIRST FIVE ESTIMATION METHODS
C
      REAL INV(MXX,MXY), Y(MXX,MXY), XEB(MXX)
      REAL XTB(MXX), VXTB(MXX), A(MXX)
C
      CALL EBTS1(INV,Y,NRC,NYR,XTB,VXTB,XEB,MXX,MXY)
      PRINT *, 'COMPLETED EBTS1'
C
      CALL EBTS2(INV,Y,NRC,NYR,XTB,VXTB,XEB,MXX,MXY)
      PRINT *, 'COMPLETED EBTS2'
C
      CALL EBOS1(INV,Y,NRC,NYR,XTB,VXTB,XEB,MXX,MXY)
      PRINT *, 'COMPLETED EBOS1'
C
      CALL EBOS2(INV,Y,NRC,NYR,XTB,VXTB,XEB,MXX,MXY)
      PRINT *, 'COMPLETED EBOS2'
C
      CALL EMTS(INV,Y,NRC,NYR,XTB,VXTB,XEB,A,MXX,MXY)
      PRINT *, 'COMPLETED EMTS'
C
      END
C
************************************************************************
C
      SUBROUTINE EBTS1(INV,Y,NRC,NYR,XTB,VXTB,XEB,MXX,MXY)
C --- TRANSFORMED SCALE, TIME INDEPENDENT VARIANCE METHOD
      REAL INV(MXX,MXY), Y(MXX,MXY), XEB(MXX)
      REAL XTB(MXX), VXTB(MXX)
      REAL MAXL,MINL,L,MAXCHI,MINCHI
      INTEGER T, VYR
      DATA AA/1.6835/, B1/-.8934/, B2/.9881/
      MAXL= -1000.0
      MINL= 1000.0
      SUML= 0.0
      KLSUM=0
      MAXCHI= -1000.0
      MINCHI= 1000.0
      SUMCHI= 0.0
      KSUM=0
      SUMMAD=0.0
      KPSUM=0
      WRITE(11,32)'  '
      WRITE(11,21)'EMP BAYES TRANS SCALE - TIME DEP VAR:'
      WRITE(11,21)'MEAN ABSOLUTE DEVIATION (ORIG SCALE):'
      WRITE(11,28)'FRACTION CELLS','FRACTION  MAD'
      WRITE(11,29)'VALID YR','K','WITH  UNDERAGE','FROM UNDERAGE','MAD'
      WRITE(11,30)
C
C --- LOOP THROUGH VALIDATION YEARS
      DO 280  VYR=1, NYR
C --- LOOP THROUGH CELLS
      DO 260  IN=1, NRC
```

```
        T=0
        SUMXT=0
        SUMVAR=0
C ---   LOOP THROUGH YEARS OF DATA TO COMPUTE XTB AND VAR(XTB)
        DO 200  IT=1, NYR
          IF(IT .NE. VYR) THEN
            IF(INV(IN,IT) .NE. 0) THEN
              X=FTT(INV(IN,IT), Y(IN,IT))
              C=SQRT(0.5+INV(IN,IT))
              XX=X+C*(3.141592654/2.0)
              XT=X/C
              T=T+1
              SUMXT=SUMXT+XT
              IF(XX .LT. 1.001) XX=1.001
              VARX=AA*(XX**B1)*(XX-1)**B2
              IF(VARX .GT. 1.0) VARX=1.0
              VARXT=VARX/(0.5+INV(IN,IT))
              SUMVAR=SUMVAR+VARXT
            ENDIF
          ENDIF
  200   CONTINUE
        XTB(IN)=SUMXT/T
        VXTB(IN)=SUMVAR/T**2
  260 CONTINUE
C
C --- CONDUCT ALGORITHM TO FIND XEB
      CALL EBITER(NRC,XTB,VXTB,XEB,MXX,VYR)
C
C --- COMPUTE MEAN SQUARED ERROR
      CALL MSE(INV,Y,NRC,NYR,VYR,XEB,L,MXX,MXY,KL)
         IF(L .LT. MINL) THEN
           MINL=L
           MINLK=KL
           MINLYR=VYR
          ELSE IF(L .GT. MAXL) THEN
           MAXL=L
           MAXLK=KL
           MAXLYR=VYR
         ENDIF
         SUML=SUML+L*KL
         KLSUM=KLSUM+KL
C
C --- INVERT XEB TO ORIGINAL SCALE
      CALL INVERT(NRC,XEB,MXX)
C
C --- COMPUTE MAD AND CHI SQUARE
      CALL OSMOE(INV,Y,NRC,NYR,VYR,XEB,CHI,K,MXX,MXY,
     *           FCELLU,FMADU,PMAD,KP)
         IF(CHI .LT. MINCHI) THEN
           MINCHI=CHI
           MNCHIK=K
           MNCHYR=VYR
          ELSE IF(CHI .GT. MAXCHI) THEN
           MAXCHI=CHI
           MXCHIK=K
           MXCHYR=VYR
```

```
          ENDIF
          SUMCHI=SUMCHI+CHI*K
          KSUM=KSUM+K
          KPSUM=KPSUM+KP
          SUMMAD=SUMMAD+PMAD*KP
       WRITE(11,31) VYR,KP,FCELLU,FMADU,PMAD
  280 CONTINUE
C
C --- WRITE RESULTS TO OUTPUT FILE
      AVGL=SUML/KLSUM
      AVGCHI=SUMCHI/KSUM
      AVGMAD=SUMMAD/KPSUM
      WRITE(11,19)'AVG MAD = ',AVGMAD
      WRITE(11,21)'CHI SQUARE (ORIG SCALE):'
      WRITE(11,26)'MIN CHI = ',MINCHI,'K = ',MNCHIK,'VALID YR = ',MNCHYR
      WRITE(11,26)'MAX CHI = ',MAXCHI,'K = ',MXCHIK,'VALID YR = ',MXCHYR
      WRITE(11,26)'AVG CHI = ',AVGCHI
      WRITE(11,21)'MEAN SQUARED ERROR (TRANS SCALE):'
      WRITE(11,25)'MIN MSE = ',MINL,'K = ',MINLK,'VALID YR = ',MINLYR
      WRITE(11,25)'MAX MSE = ',MAXL,'K = ',MAXLK,'VALID YR = ',MAXLYR
      WRITE(11,27)'AVG MSE = ',AVGL
   19 FORMAT(38X,A,F5.3)
   21 FORMAT(/1X,A)
   25 FORMAT(1X,A,F6.3,5X,A,I3,5X,A,I2)
   26 FORMAT(1X,A,F9.3,5X,A,I3,5X,A,I2)
   27 FORMAT(1X,A,F6.3/)
   28 FORMAT(17X,A,2X,A)
   29 FORMAT(1X,A,4X,A,3X,A,2X,A,3X,A)
   30 FORMAT(1X,8('-'),2X,4('-'),2X,14('-'),2X,13('-'),2X,5('-'))
   31 FORMAT(1X,I5,I8,8X,F5.3,10X,F5.3,6X,F5.3)
   32 FORMAT(1X,A)
      RETURN
      END
C
************************************************************************
C
      SUBROUTINE EBTS2(INV,Y,NRC,NYR,XTB,VXTB,XEB,MXX,MXY)
C --- TRANSFORMED SCALE, TIME INDEPENDENT VARIANCE METHOD
      REAL INV(MXX,MXY), Y(MXX,MXY), XEB(MXX)
      REAL XTB(MXX), VXTB(MXX)
      REAL MAXL,MINL,L,MAXCHI,MINCHI
      INTEGER T, VYR
      MAXL= -1000.0
      MINL= 1000.0
      SUML= 0.0
      KLSUM=0
      MAXCHI= -1000.0
      MINCHI= 1000.0
      SUMCHI= 0.0
      KSUM=0
      SUMMAD=0.0
      KPSUM=0
      WRITE(11,21)'EMP BAYES TRANS SCALE - TIME INDEP VAR:'
      WRITE(11,21)'MEAN ABSOLUTE DEVIATION (ORIG SCALE):'
      WRITE(11,28)'FRACTION CELLS','FRACTION  MAD'
      WRITE(11,29)'VALID YR','K','WITH  UNDERAGE','FROM UNDERAGE','MAD'
```

```
      WRITE(11,30)
C
C --- LOOP THROUGH VALIDATION YEARS
      DO 380  VYR=1, NYR
C --- LOOP THROUGH CELLS
      DO 360  IN=1, NRC
        T=0
        SUMXT=0
        SUMXT2=0
C ---   LOOP THROUGH YEARS OF DATA TO COMPUTE XTB AND VAR(XTB)
        DO 300  IT=1, NYR
          IF(IT .NE. VYR) THEN
            IF(INV(IN,IT) .NE. 0) THEN
              X=FTT(INV(IN,IT), Y(IN,IT))
              XT=X/SQRT(0.5+INV(IN,IT))
              T=T+1
              SUMXT=SUMXT+XT
              SUMXT2=SUMXT2+XT**2
            ENDIF
          ENDIF
  300   CONTINUE
        XTB(IN)=SUMXT/T
        VXTB(IN)=((T*SUMXT2)-(SUMXT**2))/((T-1)*T**2)
  360 CONTINUE
C
C --- CONDUCT ALGORITHM TO FIND XEB
      CALL EBITER(NRC,XTB,VXTB,XEB,MXX,VYR)
C
C --- COMPUTE MEAN SQUARED ERROR
      CALL MSE(INV,Y,NRC,NYR,VYR,XEB,L,MXX,MXY,KL)
         IF(L .LT. MINL) THEN
           MINL=L
           MINLK=KL
           MINLYR=VYR
          ELSE IF(L .GT. MAXL) THEN
           MAXL=L
           MAXLK=KL
           MAXLYR=VYR
         ENDIF
         SUML=SUML+L*KL
         KLSUM=KLSUM+KL
C
C --- INVERT XEB TO ORIGINAL SCALE
      CALL INVERT(NRC,XEB,MXX)
C
C --- COMPUTE MAD AND CHI SQUARE
      CALL OSMOE(INV,Y,NRC,NYR,VYR,XEB,CHI,K,MXX,MXY,
     *           FCELLU,FMADU,PMAD,KP)
         IF(CHI .LT. MINCHI) THEN
           MINCHI=CHI
           MNCHIK=K
           MNCHYR=VYR
          ELSE IF(CHI .GT. MAXCHI) THEN
           MAXCHI=CHI
           MXCHIK=K
           MXCHYR=VYR
```

```
         ENDIF                                                          MC802220
         SUMCHI=SUMCHI+CHI*K                                            MC802230
         KSUM=KSUM+K                                                    MC802240
         KPSUM=KPSUM+KP                                                 MC802250
         SUMMAD=SUMMAD+PMAD*KP                                          MC802260
      WRITE(11,31) VYR,KP,FCELLU,FMADU,PMAD                             MC802270
  380 CONTINUE                                                          MC802280
C                                                                       MC802290
C --- WRITE OUTPUT TO FILE                                              MC802300
      AVGL=SUML/KLSUM                                                   MC802310
      AVGCHI=SUMCHI/KSUM                                                MC802320
      AVGMAD=SUMMAD/KPSUM                                               MC802330
      WRITE(11,19)'AVG MAD = ',AVGMAD                                   MC802340
      WRITE(11,21)'CHI SQUARE (ORIG SCALE):'                            MC802350
      WRITE(11,26)'MIN CHI = ',MINCHI,'K = ',MNCHIK,'VALID YR = ',MNCHYRMC802360
      WRITE(11,26)'MAX CHI = ',MAXCHI,'K = ',MXCHIK,'VALID YR = ',MXCHYRMC802370
      WRITE(11,26)'AVG CHI = ',AVGCHI                                   MC802380
      WRITE(11,21)'MEAN SQUARED ERROR (TRANS SCALE):'                   MC802390
      WRITE(11,25)'MIN MSE = ',MINL,'K = ',MINLK,'VALID YR = ',MINLYR   MC802400
      WRITE(11,25)'MAX MSE = ',MAXL,'K = ',MAXLK,'VALID YR = ',MAXLYR   MC802410
      WRITE(11,27)'AVG MSE = ',AVGL                                     MC802420
   19 FORMAT(38X,A,F5.3)                                                MC802430
   21 FORMAT(/1X,A)                                                     MC802440
   25 FORMAT(1X,A,F6.3,5X,A,I3,5X,A,I2)                                 MC802450
   26 FORMAT(1X,A,F9.3,5X,A,I3,5X,A,I2)                                 MC802460
   27 FORMAT(1X,A,F6.3/)                                                MC802470
   28 FORMAT(17X,A,2X,A)                                                MC802480
   29 FORMAT(1X,A,4X,A,3X,A,2X,A,3X,A)                                  MC802490
   30 FORMAT(1X,8('-'),2X,4('-'),2X,14('-'),2X,13('-'),2X,5('-'))       MC802500
   31 FORMAT(1X,I5,I8,8X,F5.3,10X,F5.3,6X,F5.3)                         MC802510
      RETURN                                                            MC802520
      END                                                               MC802530
C                                                                       MC802540
************************************************************************MC802550
C                                                                       MC802560
      SUBROUTINE EBOS1(INV,Y,NRC,NYR,XTB,VXTB,XEB,MXX,MXY)              MC802570
C --- ORIGINAL SCALE, TIME DEPENDENT VARIANCE METHOD                    MC802580
      REAL INV(MXX,MXY), Y(MXX,MXY), XEB(MXX)                           MC802590
      REAL XTB(MXX), VXTB(MXX)                                          MC802600
      REAL MAXCHI, MINCHI                                               MC802610
      INTEGER T, VYR                                                    MC802620
      MAXCHI= -1000.0                                                   MC802630
      MINCHI= 1000.0                                                    MC802640
      SUMCHI= 0.0                                                       MC802650
      KSUM=0                                                            MC802660
      SUMMAD=0.0                                                        MC802670
      KPSUM=0                                                           MC802680
      WRITE(11,21)'EMP BAYES ORIG SCALE - TIME DEP VAR:'                MC802690
      WRITE(11,21)'MEAN ABSOLUTE DEVIATION (ORIG SCALE):'               MC802700
      WRITE(11,28)'FRACTION CELLS','FRACTION  MAD'                      MC802710
      WRITE(11,29)'VALID YR','K','WITH  UNDERAGE','FROM UNDERAGE','MAD' MC802720
      WRITE(11,30)                                                      MC802730
C                                                                       MC802740
C --- LOOP THROUGH VALIDATION YEARS                                     MC802750
      DO 480  VYR=1, NYR                                                MC802760
C --- LOOP THROUGH CELLS                                                MC802770
```

```
      DO 460  IN=1, NRC
        T=0
        SUMXT=0
        SUMVAR=0
C ---   LOOP THROUGH YEARS OF DATA TO COMPUTE XTB AND VAR(XTB)
        DO 400  IT=1, NYR
          IF(IT .NE. VYR) THEN
            IF(INV(IN,IT) .NE. 0) THEN
              PHAT=Y(IN,IT)/INV(IN,IT)
              T=T+1
              SUMXT=SUMXT+PHAT
              IF(PHAT .GT. 0.0) THEN
                SUMVAR=SUMVAR+PHAT*(1-PHAT)/INV(IN,IT)
              ELSE
                SUMVAR=SUMVAR+1/(INV(IN,IT)+1)**2
              ENDIF
            ENDIF
          ENDIF
  400   CONTINUE
        XTB(IN)=SUMXT/T
        VXTB(IN)=SUMVAR/T**2
  460 CONTINUE
C
C --- CONDUCT ALGORITHM TO FIND XEB
      CALL EBITER(NRC,XTB,VXTB,XEB,MXX,VYR)
C
C --- COMPUTE MAD AND CHI SQUARE
      CALL OSMOE(INV,Y,NRC,NYR,VYR,XEB,CHI,K,MXX,MXY,
     *           FCELLU,FMADU,PMAD,KP)
         IF(CHI .LT. MINCHI) THEN
           MINCHI=CHI
           MNCHIK=K
           MNCHYR=VYR
          ELSE IF(CHI .GT. MAXCHI) THEN
           MAXCHI=CHI
           MXCHIK=K
           MXCHYR=VYR
         ENDIF
         SUMCHI=SUMCHI+CHI*K
         KSUM=KSUM+K
         KPSUM=KPSUM+KP
         SUMMAD=SUMMAD+PMAD*KP
      WRITE(11,31) VYR,KP,FCELLU,FMADU,PMAD
  480 CONTINUE
C
C --- WRITE OUTPUT TO FILE
      AVGCHI=SUMCHI/KSUM
      AVGMAD=SUMMAD/KPSUM
      WRITE(11,19)'AVG MAD = ',AVGMAD
      WRITE(11,21)'CHI SQUARE (ORIG SCALE):'
      WRITE(11,26)'MIN CHI = ',MINCHI,'K = ',MNCHIK,'VALID YR = ',MNCHYR
      WRITE(11,26)'MAX CHI = ',MAXCHI,'K = ',MXCHIK,'VALID YR = ',MXCHYR
      WRITE(11,27)'AVG CHI = ',AVGCHI
  19  FORMAT(38X,A,F5.3)
  21  FORMAT(/1X,A)
  26  FORMAT(1X,A,F9.3,5X,A,I3,5X,A,I2)
```

```
   27  FORMAT(1X,A,F9.3/)
   28  FORMAT(17X,A,2X,A)
   29  FORMAT(1X,A,4X,A,3X,A,2X,A,3X,A)
   30  FORMAT(1X,8('-'),2X,4('-'),2X,14('-'),2X,13('-'),2X,5('-'))
   31  FORMAT(1X,I5,I8,8X,F5.3,10X,F5.3,6X,F5.3)
       RETURN
       END
C
************************************************************************
C
       SUBROUTINE EBOS2(INV,Y,NRC,NYR,XTB,VXTB,XEB,MXX,MXY)
C ---  ORIGINAL SCALE, TIME INDEPENDENT VARIANCE METHOD
       REAL INV(MXX,MXY), Y(MXX,MXY), XEB(MXX)
       REAL XTB(MXX), VXTB(MXX)
       REAL MAXCHI, MINCHI
       INTEGER T, VYR
       MAXCHI= -1000.0
       MINCHI= 1000.0
       SUMCHI= 0.0
       KSUM=0
       SUMMAD=0.0
       KPSUM=0
       WRITE(11,21)'EMP BAYES ORIG SCALE - TIME INDEP VAR:'
       WRITE(11,21)'MEAN ABSOLUTE DEVIATION (ORIG SCALE):'
       WRITE(11,28)'FRACTION CELLS','FRACTION  MAD'
       WRITE(11,29)'VALID YR','K','WITH  UNDERAGE','FROM UNDERAGE','MAD'
       WRITE(11,30)
C
C ---  LOOP THROUGH VALIDATION YEARS
       DO 580  VYR=1, NYR
C ---  LOOP THROUGH CELLS
       DO 560  IN=1, NRC
         T=0
         SUMXT=0
         SUMVAR=0
         SUMY=0
         SUMINV=0
C ---    LOOP THROUGH YEARS OF DATA TO COMPUTE XTB AND VAR(XTB)
         DO 500  IT=1, NYR
           IF(IT .NE. VYR) THEN
             IF(INV(IN,IT) .NE. 0) THEN
               PHAT=Y(IN,IT)/INV(IN,IT)
               SUMXT=SUMXT+PHAT
               SUMY=SUMY+Y(IN,IT)
               SUMINV=SUMINV+INV(IN,IT)
               T=T+1
               SUMVAR=SUMVAR+1.0/INV(IN,IT)
             ENDIF
           ENDIF
  500    CONTINUE
         XTB(IN)=SUMXT/T
         IF(SUMY .GT. 0.0) THEN
           PTILDE=SUMY/SUMINV
           VXTB(IN)=(PTILDE*(1-PTILDE)*SUMVAR)/T**2
         ELSE
           VXTB(IN)=SUMINV*SUMVAR/(((1+SUMINV)**2)*T**2)
```

```
        ENDIF
  560 CONTINUE
C
C --- CONDUCT ALGORITHM TO FIND XEB
      CALL EBITER(NRC,XTB,VXTB,XEB,MXX,VYR)
C
C --- COMPUTE MAD AND CHI SQUARE
      CALL OSMOE(INV,Y,NRC,NYR,VYR,XEB,CHI,K,MXX,MXY,
     *           FCELLU,FMADU,PMAD,KP)
         IF(CHI .LT. MINCHI) THEN
           MINCHI=CHI
           MNCHIK=K
           MNCHYR=VYR
          ELSE IF(CHI .GT. MAXCHI) THEN
           MAXCHI=CHI
           MXCHIK=K
           MXCHYR=VYR
         ENDIF
         SUMCHI=SUMCHI+CHI*K
         KSUM=KSUM+K
         KPSUM=KPSUM+KP
         SUMMAD=SUMMAD+PMAD*KP
      WRITE(11,31) VYR,KP,FCELLU,FMADU,PMAD
  580 CONTINUE
C
C --- WRITE OUTPUT TO FILE
      AVGCHI=SUMCHI/KSUM
      AVGMAD=SUMMAD/KPSUM
      WRITE(11,19)'AVG MAD = ',AVGMAD
      WRITE(11,21)'CHI SQUARE (ORIG SCALE):'
      WRITE(11,26)'MIN CHI = ',MINCHI,'K = ',MNCHIK,'VALID YR = ',MNCHYR
      WRITE(11,26)'MAX CHI = ',MAXCHI,'K = ',MXCHIK,'VALID YR = ',MXCHYR
      WRITE(11,27)'AVG CHI = ',AVGCHI
   19 FORMAT(38X,A,F5.3)
   21 FORMAT(/1X,A)
   26 FORMAT(1X,A,F9.3,5X,A,I3,5X,A,I2)
   27 FORMAT(1X,A,F9.3/)
   28 FORMAT(17X,A,2X,A)
   29 FORMAT(1X,A,4X,A,3X,A,2X,A,3X,A)
   30 FORMAT(1X,8('-'),2X,4('-'),2X,14('-'),2X,13('-'),2X,5('-'))
   31 FORMAT(1X,I5,I8,8X,F5.3,10X,F5.3,6X,F5.3)
      RETURN
      END
C
************************************************************************
C
      SUBROUTINE EMTS(INV,Y,NRC,NYR,XTB,VXTB,XEB,A,MXX,MXY)
C --- EFRON-MORRIS METHOD
      REAL INV(MXX,MXY), Y(MXX,MXY), XEB(MXX)
      REAL XTB(MXX), VXTB(MXX), A(MXX)
      REAL MAXL,MINL,L,MAXCHI,MINCHI
      INTEGER T, VYR
      DATA AA/1.6835/, B1/-.8934/, B2/.9881/
      MAXL= -1000.0
      MINL= 1000.0
      SUML= 0.0
```

```
      KLSUM=0
      MAXCHI= -1000.0
      MINCHI= 1000.0
      SUMCHI= 0.0
      KSUM=0
      SUMMAD=0.0
      KPSUM=0
      WRITE(11,21)'EFRON-MORRIS TRANS SCALE - TIME DEP VAR:'
      WRITE(11,21)'MEAN ABSOLUTE DEVIATION (ORIG SCALE):'
      WRITE(11,28)'FRACTION CELLS','FRACTION  MAD'
      WRITE(11,29)'VALID YR','K','WITH  UNDERAGE','FROM UNDERAGE','MAD'
      WRITE(11,30)
C
C --- LOOP THROUGH VALIDATION YEARS
      DO 280  VYR=1, NYR
C --- LOOP THROUGH CELLS
      DO 260  IN=1, NRC
        T=0
        SUMXT=0
        SUMVAR=0
C ---   LOOP THROUGH YEARS OF DATA TO COMPUTE XTB AND VAR(XTB)
        DO 200  IT=1, NYR
          IF(IT .NE. VYR) THEN
            IF(INV(IN,IT) .NE. 0) THEN
              X=FTT(INV(IN,IT), Y(IN,IT))
              C=SQRT(0.5+INV(IN,IT))
              XX=X+C*(3.141592654/2.0)
              XT=X/C
              T=T+1
              SUMXT=SUMXT+XT
              IF(XX .LT. 1.001) XX=1.001
              VARX=AA*(XX**B1)*(XX-1)**B2
              IF(VARX .GT. 1.0) VARX=1.0
              VARXT=VARX/(0.5+INV(IN,IT))
              SUMVAR=SUMVAR+VARXT
            ENDIF
          ENDIF
  200   CONTINUE
        XTB(IN)=SUMXT/T
        VXTB(IN)=SUMVAR/T**2
  260 CONTINUE
C
C --- CONDUCT ALGORITHM TO FIND XEB
      CALL EMITER(NRC,XTB,VXTB,XEB,A,MXX,VYR)
C
C --- COMPUTE MEAN SQUARED ERROR
      CALL MSE(INV,Y,NRC,NYR,VYR,XEB,L,MXX,MXY,KL)
         IF(L .LT. MINL) THEN
           MINL=L
           MINLK=KL
           MINLYR=VYR
          ELSE IF(L .GT. MAXL) THEN
           MAXL=L
           MAXLK=KL
           MAXLYR=VYR
         ENDIF
```

```
         SUML=SUML+L*KL
         KLSUM=KLSUM+KL
C
C --- INVERT XEB TO ORIGINAL SCALE
      CALL INVERT(NRC,XEB,MXX)
C
C --- COMPUTE MAD AND CHI SQUARE
      CALL OSMOE(INV,Y,NRC,NYR,VYR,XEB,CHI,K,MXX,MXY,
     *           FCELLU,FMADU,PMAD,KP)
         IF(CHI .LT. MINCHI) THEN
           MINCHI=CHI
           MNCHIK=K
           MNCHYR=VYR
          ELSE IF(CHI .GT. MAXCHI) THEN
           MAXCHI=CHI
           MXCHIK=K
           MXCHYR=VYR
         ENDIF
         SUMCHI=SUMCHI+CHI*K
         KSUM=KSUM+K
         KPSUM=KPSUM+KP
         SUMMAD=SUMMAD+PMAD*KP
      WRITE(11,31) VYR,KP,FCELLU,FMADU,PMAD
  280 CONTINUE
C
C --- WRITE OUTPUT TO FILE
      AVGL=SUML/KLSUM
      AVGCHI=SUMCHI/KSUM
      AVGMAD=SUMMAD/KPSUM
      WRITE(11,19)'AVG MAD = ',AVGMAD
      WRITE(11,21)'CHI SQUARE (ORIG SCALE):'
      WRITE(11,26)'MIN CHI = ',MINCHI,'K = ',MNCHIK,'VALID YR = ',MNCHYR
      WRITE(11,26)'MAX CHI = ',MAXCHI,'K = ',MXCHIK,'VALID YR = ',MXCHYR
      WRITE(11,26)'AVG CHI = ',AVGCHI
      WRITE(11,21)'MEAN SQUARED ERROR (TRANS SCALE):'
      WRITE(11,25)'MIN MSE = ',MINL,'K = ',MINLK,'VALID YR = ',MINLYR
      WRITE(11,25)'MAX MSE = ',MAXL,'K = ',MAXLK,'VALID YR = ',MAXLYR
      WRITE(11,27)'AVG MSE = ',AVGL
   19 FORMAT(38X,A,F5.3)
   21 FORMAT(/1X,A)
   25 FORMAT(1X,A,F6.3,5X,A,I3,5X,A,I2)
   26 FORMAT(1X,A,F9.3,5X,A,I3,5X,A,I2)
   27 FORMAT(1X,A,F6.3)
   28 FORMAT(17X,A,2X,A)
   29 FORMAT(1X,A,4X,A,3X,A,2X,A,3X,A)
   30 FORMAT(1X,8('-'),2X,4('-'),2X,14('-'),2X,13('-'),2X,5('-'))
   31 FORMAT(1X,I5,I8,8X,F5.3,10X,F5.3,6X,F5.3)
      RETURN
      END
C
************************************************************************
C
      SUBROUTINE EBITER(NRC,XTB,VXTB,XEB,MXX,VYR)
C --- ITERATIVE ALGORITHM TO SOLVE FOR XEB
      REAL XTB(MXX), VXTB(MXX), XEB(MXX)
      INTEGER VYR
```

```
      A=0
      ITER=0
  100 CONTINUE
      ITER=ITER+1
      IF(ITER .GT. 100) PRINT *,'EBITER GT 100'
      A0=A
      SUMALK=0
C
C --- SUM THE ALPHAS
      DO 200  I=1,NRC
        SUMALK=SUMALK+1/(A+VXTB(I))
  200 CONTINUE
C
C --- COMPUTE XBB
      XBB=0
      DO 300  I=1,NRC
        ALPHA=1/(A+VXTB(I))
        GAMMA=ALPHA/SUMALK
        XBB=XBB+GAMMA*XTB(I)
  300 CONTINUE
C
C --- UPDATE VALUE OF A
      SUMNUM=0
      SUMDEN=0
      DO 400  I=1,NRC
        ALPHA=1/(A+VXTB(I))
        SUMNUM=SUMNUM+ALPHA*(XTB(I)-XBB)**2
        SUMDEN=SUMDEN+((XTB(I)-XBB)**2)*ALPHA**2
  400 CONTINUE
      A=A-(NRC-1-SUMNUM)/SUMDEN
      IF(A .LE. 0) THEN
        A=0
        GO TO 500
      ENDIF
      IF(ABS(A-A0) .GT. 0.0001) GO TO 100
  500 CONTINUE
C
C --- ITERATIONS CONVERGED, COMPUTE XEB
      DO 600  I=1,NRC
        X=XTB(I)
        V=VXTB(I)
        XEB(I)=(A*X)/(A+V)+(V*XBB)/(A+V)
  600 CONTINUE
      RETURN
      END
C
************************************************************************
C
      SUBROUTINE EMITER(NRC,XTB,VXTB,XEM,A,MXX,VYR)
C --- ITERATIVE ALGORITHM TO SOLVE FOR XEB FOR EFRON-MORRIS METHOD
      REAL XTB(MXX), VXTB(MXX), XEM(MXX), A(MXX)
      INTEGER VYR
      DO 100  I=1,NRC
        A(I)=0
```

```
  100 CONTINUE
C
C --- SUM THE ALPHAS
      SUMALK=0
      DO 200  I=1,NRC
        SUMALK=SUMALK+1/(A(I)+VXTB(I))
  200 CONTINUE
C
C --- COMPUTE XHAT
      XHAT=0
      DO 300  I=1,NRC
        ALPHA=1/(A(I)+VXTB(I))
        GAMMA=ALPHA/SUMALK
        XHAT=XHAT+GAMMA*XTB(I)
  300 CONTINUE
  311 XHATP=XHAT
      I=1
  333 AP=A(I)
      S=(XTB(I)-XHAT)**2
C
C --- COMPUTE SN AND SD
      SN=0
      SD=0
      DO 400  J=1,NRC
        IF(J .NE. I) THEN
          DEN=(A(J)+VXTB(J))**2
          SN=SN+((XTB(J)-XHAT)**2-VXTB(J))/DEN
          SD=SD+1/DEN
        ENDIF
  400 CONTINUE
C
C --- NEWTON-RAPHSON ITERATIONS TO SOLVE FOR A
  444 AD=A(I)+VXTB(I)
      GNUM=S-3*VXTB(I)+SN*AD**2
      GDEN=3+SD*AD**2
      GPRM=2*AD*SN/GDEN-2*GNUM*AD*SD/GDEN**2
      G=GNUM/GDEN
      A(I)=A(I)-((A(I)-G)/(1-GPRM))
      IF(A(I) .LE. 0.) THEN
        A(I)=0.0
        I=I+1
        IF(I .LE. NRC) THEN
          GO TO 333
         ELSE
          GO TO 555
        ENDIF
      ENDIF
      IF(ABS(A(I)-AP) .LE. 0.0001) THEN
        I=I+1
        IF(I .LE. NRC) THEN
          GO TO 333
         ELSE
          GO TO 555
        ENDIF
       ELSE
        AP=A(I)
```

```
         GO TO 444
      ENDIF
C
C --- TEST FOR CONVERGENCE: ABS(S-SP) LT EPSILON
  555 SUMALK=0
      DO 600  J=1,NRC
        SUMALK=SUMALK+1/(A(J)+VXTB(J))
  600 CONTINUE
      XHAT=0
      DO 700  J=1,NRC
        ALPHA=1/(A(J)+VXTB(J))
        GAMMA=ALPHA/SUMALK
        XHAT=XHAT+GAMMA*XTB(J)
  700 CONTINUE
      DO 800  J=1,NRC
        S=(XTB(J)-XHAT)**2
        SP=(XTB(J)-XHATP)**2
        IF(ABS(S-SP) .GT. 0.0001) GO TO 311
  800 CONTINUE
C
C --- ITERATIONS CONVERGED, COMPUTE XEM
      DO 950  K=1,NRC
        SD=0
        DO 900  J=1,NRC
          IF(J .NE. K) SD=SD+1/(A(J)+VXTB(J))**2
  900   CONTINUE
        AD=A(K)+VXTB(K)
        DSTAR=3+(AD**2)*SD
        B=(1.-4./DSTAR)*VXTB(K)/AD
        IF(B .GT. 1.0) B=1.0
        IF(B .LT. 0.0) B=0.0
        XEM(K)=XHAT+(1-B)*(XTB(K)-XHAT)
  950 CONTINUE
      END
C
************************************************************************
C
      SUBROUTINE MSE(INV,Y,NRC,NYR,VYR,XEB,L,MXX,MXY,KL)
C --- COMPUTES MEAN SQUARED ERROR MOE
      REAL INV(MXX,MXY), Y(MXX,MXY), XEB(MXX)
      REAL L,MU
      INTEGER VYR
      SUMSE=0
      KL=0
      DO 100  I=1,NRC
        IF(INV(I,VYR) .GT. 0.0) THEN
          X=FTT(INV(I,VYR),Y(I,VYR))
          MU=X/SQRT(0.5+INV(I,VYR))
          SUMSE=SUMSE+(XEB(I)-MU)**2
          KL=KL+1
        ENDIF
  100 CONTINUE
      L=SUMSE/KL
      RETURN
      END
C
```

```
*****************************************************************************
C
      SUBROUTINE OSMOE(INV,Y,NRC,NYR,VYR,XEB,CHI,K,MXX,MXY,
     *                 FCELLU,FMADU,PMAD,KP)
C --- COMPUTES MAD AND CHI SQUARE MOES
      REAL INV(MXX,MXY), Y(MXX,MXY), XEB(MXX)
      INTEGER VYR
      CHI=0.0
      K=0
      SUMPMO=0.0
      SUMPMU=0.0
      KPMO=0
      KPMU=0
      DO 100  I=1,NRC
        P=XEB(I)
        E=P*INV(I,VYR)
        A=Y(I,VYR)
C
C ---   COMPUTE MAD FOR THIS CELL
        IF(INV(I,VYR) .GT.  0.0) THEN
          PA=A/INV(I,VYR)
          IF(P .GT.  PA) THEN
            SUMPMO=SUMPMO+(P-PA)
            KPMO=KPMO+1
          ELSE
            SUMPMU=SUMPMU+(PA-P)
            KPMU=KPMU+1
          ENDIF
        ENDIF
C
C ---   COMPUTE CHI SQUARE FOR THIS CELL
        IF(E .NE.  0.0  .AND.  P .NE.  1.0) THEN
          K=K+1
          CHI=CHI+((A-E)**2)/(E*(1-P))
        ENDIF
  100 CONTINUE
C
C --- COMPUTE WEIGHTED AVERAGES
      KP=KPMO+KPMU
      FCELLU=REAL(KPMU)/REAL(KP)
      FMADU=SUMPMU/(SUMPMU+SUMPMO)
      PMAD=(SUMPMU+SUMPMO)/KP
      RETURN
      END
C
*****************************************************************************
C
      SUBROUTINE INVERT(NRC,XEB,MXX)
C --- INVERT XEB TO ORIGINAL SCALE
      REAL XEB(MXX)
      DO 100  I=1, NRC
        P=0.5*(1+SIN(XEB(I)))
        IF (P .LT.  0.0) THEN
          P=0.0
         ELSE IF (P .GT.  1.0) THEN
          P=1.0
```

```
          ENDIF
          XEB(I)=P
  100 CONTINUE
      RETURN
      END
C
************************************************************************
C
      FUNCTION FTT(INV,Y)
C --- CONDUCTS FREMAN-TUKEY TRANSFORM
      REAL INV,Y
      TEMP =-1. + 2.*Y/(1.+INV)
      TEMP1=-1. + 2.*(1.+Y)/(1.+INV)
      IF(ABS(TEMP).GT.1 .OR. ABS(TEMP1).GT.1) THEN
              WRITE(6,*) 'FTT ERROR INV,Y=',INV,Y,TEMP,TEMP1
              FTT=1
              RETURN
      ENDIF
      FTT=SQRT(.5+INV)*.5*(ASIN(TEMP) + ASIN(TEMP1))
      END
```

D. VECTOR METHOD SUBROUTINE

```
      SUBROUTINE MC87V(INV,Y,MXX,NYR,NRC,XTBJI,DELTA,X,XVYR,VYRINV,
     *  VYRY,BSTAR,S,GAMMA,XBBJ,EVAL,MXP,MXK,BKTBL,NBK,NSC,NCSR,ISFLAG)
C --- VECTOR METHOD
      REAL INV(MXX,NYR), Y(MXX,NYR)
      REAL XTBJI(MXP,MXK), DELTA(MXP,MXK), X(MXP,MXK)
      REAL XVYR(MXP,MXK), VYRINV(MXP,MXK), VYRY(MXP,MXK)
      REAL BSTAR(MXP,MXP), S(MXP,MXP), GAMMA(MXP,MXP)
      REAL XBBJ(MXP), EVAL(MXP)
      INTEGER*2 BKTBL(MXX,3)
C
      REAL MAXL,MINL,L,MAXCHI,MINCHI,MO,MU,MAD
      INTEGER T, VYR, P
C
      MAXL= -1000.0
      MINL= 1000.0
      SUML= 0.0
      KPSUM=0
      MAXCHI= -1000.0
      MINCHI= 1000.0
      SUMCHI= 0.0
      KCSUM=0
      WRITE(11,32)'  '
      WRITE(11,21)'EMP BAYES TRANS SCALE - VECTOR CASE:'
      IF (ISFLAG .EQ. 1) THEN
        P=NSC
        WRITE(11,21)'VECTOR IS BY SERVICE COMPONENT'
       ELSE
        P=NCSR
        WRITE(11,21)'VECTOR IS BY COMMISSIONING SOURCE'
      ENDIF
      WRITE(11,22)'K=',NRC,'P=',P,'KP= ',(NRC*P)
      WRITE(11,21)'MEAN ABSOLUTE DEVIATION (ORIG SCALE):'
      WRITE(11,28)'FRACTION CELLS','FRACTION  MAD'
      WRITE(11,29)'VALID YR','KP','WITH  UNDERAGE','FROM UNDERAGE','MAD'
      WRITE(11,30)
      K=NRC
      IF(K .LE. (P+2)) THEN
        WRITE(6,*)'*** ERROR IN VECTOR CASE: P+2 GT K ***'
        STOP
      ENDIF
C
C --- CONDUCT VALIDATION
      DO 999  VYR=1, NYR
      DO 90  J=1,MXP
        DO 80  I=1,MXK
          XTBJI(J,I)=0.0
          DELTA(J,I)=0.0
          XVYR(J,I)=0.0
 80     CONTINUE
 90   CONTINUE
      KMKG=BKTBL(1,1)
      NRC=1
C --- LOOP THROUGH CELLS IN VECTOR FORM
```

```
      DO 130  I=1,NBK
        IF(BKTBL(I,1) .NE. KMKG) NRC=NRC+1
        DO 100  J=1,P
          IF(BKTBL(I,2) .EQ. J) THEN
            JP=J
            GO TO 110
          ENDIF
  100   CONTINUE
        WRITE(6,*) '*** ERROR IN P VECTOR ASSIGNMENT ***'
  110   T=0
        SUMXT=0
        SUMVAR=0
C ---   LOOP THROUGH YEARS OF DATA TO SOLVE FOR XTB AND VAR(XTB)
        DO 120  IT=1, NYR
          IF(IT .NE. VYR) THEN
            IF(INV(I,IT) .NE. 0) THEN
              XIJ=FTTV(INV(I,IT), Y(I,IT))
              C=0.5+INV(I,IT)
              XT=XIJ/SQRT(C)
              SUMXT=SUMXT+XT
              SUMVAR=SUMVAR+1/C
              T=T+1
            ENDIF
          ENDIF
  120   CONTINUE
        XTBJI(JP,NRC)=SUMXT/T
C ---   STORE VARIANCE MATRIX IN DELTA MATRIX (TEMPORARY)
        DELTA(JP,NRC)=SUMVAR/T**2
C ---   GET VALIDATION YEAR ESTIMATE, INVENTORY AND ATTRITION INFO
        IF(INV(I,VYR) .GT. 0.0) THEN
          XIJ=FTTV(INV(I,VYR), Y(I,VYR))
          XT=XIJ/SQRT(0.5+INV(I,VYR))
          XVYR(JP,NRC)=XT
        ENDIF
        VYRINV(JP,NRC)=INV(I,VYR)
        VYRY(JP,NRC)=Y(I,VYR)
        KMKG=BKTBL(I,1)
  130 CONTINUE
      IF(K .NE. NRC) THEN
        WRITE(6,*) '*** ERROR IN VECTOR CASE: K NE NRC ***'
      ENDIF
C
C --- COMPUTE XBB SUB J
      DO 210  J=1,P
        SUMXTB=0.0
        DO 200  I=1,K
          SUMXTB=SUMXTB+XTBJI(J,I)
  200   CONTINUE
        XBBJ(J)=SUMXTB/K
  210 CONTINUE
C
C --- COMPUTE X SUB JI MATRIX, MAKE A COPY IN DELTA MATRIX (TEMPORARY)
      DO 230  J=1,P
        DO 220  I=1,K
          X(J,I)=(XTBJI(J,I)-XBBJ(J))*SQRT(DELTA(J,I))
          DELTA(J,I)=X(J,I)
```

```
  220   CONTINUE
  230 CONTINUE
C
C --- COMPUTE S MATRIX
      CALL MXYTF(P,K,X,MXP,P,K,DELTA,MXP,P,P,S,MXP)
C
C --- DO EIGENANALYSIS OF S
C --- PUT EIGENVALUES INTO EVAL, EIGENVECTORS INTO GAMMA
      CALL EVCSF(P,S,MXP,EVAL,GAMMA,MXP)
C
C --- CREATE ESTAR INVERSE
      KP2=K-P-2
      DO 240  J=1,P
        IF(EVAL(J) .LT. KP2) THEN
          EVAL(J)=KP2
        ENDIF
        EVAL(J)=1.0/EVAL(J)
  240 CONTINUE
      DO 260  I=1,P
        DO 250  J=1,P
          IF(I .EQ. J) THEN
            BSTAR(I,J)=EVAL(J)
           ELSE
            BSTAR(I,J)=0.0
          ENDIF
  250   CONTINUE
  260 CONTINUE
C
C --- CREATE BSTAR = I - (K-P-2) S TILDE INVERSE
      CALL MRRRR(P,P,GAMMA,MXP,P,P,BSTAR,MXP,P,P,S,MXP)
      CALL MXYTF(P,P,S,MXP,P,P,GAMMA,MXP,P,P,BSTAR,MXP)
      DO 280  I=1,P
        DO 270  J=1,P
          BSTAR(I,J)=KP2*BSTAR(I,J)
          IF(I .EQ. J) THEN
            BSTAR(I,J)=1.0-BSTAR(I,J)
           ELSE
            BSTAR(I,J)=0.0-BSTAR(I,J)
          ENDIF
  270   CONTINUE
  280 CONTINUE
C
C --- COMPUTE DELTA SUB JI
      DO 300  J=1,P
        DO 290  I=1,K
          X(J,I)=XTBJI(J,I)-XBBJ(J)
  290   CONTINUE
  300 CONTINUE
      CALL MRRRR(P,P,BSTAR,MXP,P,K,X,MXP,P,K,XTBJI,MXP)
      DO 320  J=1,P
        DO 310  I=1,K
          DELTA(J,I)=XBBJ(J)+XTBJI(J,I)
  310   CONTINUE
  320 CONTINUE
C
C --- COMPUTE MSE
```

```
      KP=0
      SUMSE=0.0
      DO 340  J=1,P
        DO 330  I=1,K
          IF(INV(I,VYR) .GT. 0.0) THEN
            SUMSE=SUMSE+(DELTA(J,I)-XVYR(J,I))**2
            KP=KP+1
          ENDIF
  330   CONTINUE
  340 CONTINUE
      L=SUMSE/KP
      IF(L .LT. MINL) THEN
        MINL=L
        MINLKP=KP
        MINLYR=VYR
       ELSE IF(L .GT. MAXL) THEN
        MAXL=L
        MAXLKP=KP
        MAXLYR=VYR
      ENDIF
      SUML=SUML+L*KP
      KPSUM=KPSUM+KP
C
C --- INVERT DELTA SUB JI BACK TO ORIGINAL SCALE
      DO 360  J=1,P
        DO 350  I=1,K
          PHAT=0.5*(1+SIN(DELTA(J,I)))
          IF (PHAT .LT. 0.0) THEN
            PHAT=0.0
           ELSE IF (PHAT .GT. 1.0) THEN
            PHAT=1.0
          ENDIF
          DELTA(J,I)=PHAT
  350   CONTINUE
  360 CONTINUE
C
C --- COMPUTE CHI SQUARE AND MAD
      CHI=0.0
      KCHI=0
      SUMPMO=0.0
      SUMPMU=0.0
      KPMO=0
      KPMU=0
      DO 410  J=1,P
        DO 400  I=1,K
          PHAT=DELTA(J,I)
          E=PHAT*VYRINV(J,I)
          A=VYRY(J,I)
          IF(VYRINV(J,I) .GT. 0.0) THEN
            PACT=A/VYRINV(J,I)
            IF(PHAT .GT. PACT) THEN
              SUMPMO=SUMPMO+(PHAT-PACT)
              KPMO=KPMO+1
             ELSE
              SUMPMU=SUMPMU+(PACT-PHAT)
              KPMU=KPMU+1
```

```
            ENDIF
          ENDIF
          IF(E.NE.0.0 .AND. PHAT.NE.1.0) THEN
            KCHI=KCHI+1
            CHI=CHI+((A-E)**2)/(E*(1-PHAT))
          ENDIF
  400   CONTINUE
  410 CONTINUE
      KMAD=KPMO+KPMU
      FCELLU=REAL(KPMU)/REAL(KMAD)
      FMADU=SUMPMU/(SUMPMU+SUMPMO)
      PMAD=(SUMPMU+SUMPMO)/KMAD
      IF(CHI .LT. MINCHI) THEN
        MINCHI=CHI
        MNCHIK=KCHI
        MNCHYR=VYR
       ELSE IF(CHI .GT. MAXCHI) THEN
        MAXCHI=CHI
        MXCHIK=KCHI
        MXCHYR=VYR
      ENDIF
      SUMCHI=SUMCHI+CHI*KCHI
      KCSUM=KCSUM+KCHI
      WRITE(11,31) VYR,KMAD,FCELLU,FMADU,PMAD
  999 CONTINUE
      AVGL=SUML/KPSUM
      AVGCHI=SUMCHI/KCSUM
      WRITE(11,21)'CHI SQUARE (ORIG SCALE):'
      WRITE(11,26)'MIN CHI = ',MINCHI,'KP = ',MNCHIK,
     *            'VALID YR = ',MNCHYR
      WRITE(11,26)'MAX CHI = ',MAXCHI,'KP = ',MXCHIK,
     *            'VALID YR = ',MXCHYR
      WRITE(11,26)'AVG CHI = ',AVGCHI
      WRITE(11,21)'MEAN SQUARED ERROR (TRANS SCALE):'
      WRITE(11,25)'MIN MSE = ',MINL,'KP = ',MINLKP,'VALID YR = ',MINLYR
      WRITE(11,25)'MAX MSE = ',MAXL,'KP = ',MAXLKP,'VALID YR = ',MAXLYR
      WRITE(11,27)'AVG MSE = ',AVGL
   21 FORMAT(/1X,A)
   22 FORMAT(1X,3(A,I3,5X))
   25 FORMAT(1X,A,F6.3,5X,A,I3,5X,A,I2)
   26 FORMAT(1X,A,F9.3,5X,A,I3,5X,A,I2)
   27 FORMAT(1X,A,F6.3/)
   28 FORMAT(17X,A,2X,A)
   29 FORMAT(1X,A,3X,A,3X,A,2X,A,3X,A)
   30 FORMAT(1X,8('-'),2X,4('-'),2X,14('-'),2X,13('-'),2X,5('-'))
   31 FORMAT(1X,I5,I8,8X,F5.3,10X,F5.3,6X,F5.3)
   32 FORMAT(1X,A)
      WRITE(6,*)'COMPLETED VECTOR CASE'
      END
C
************************************************************************
C
      FUNCTION FTTV(INV,Y)
C --- CONDUCTS FREMAN-TUKEY TRANSFORM
      REAL INV,Y
      TEMP =-1. + 2.*Y/(1.+INV)
```

```
      TEMP1=-1. + 2.*(1.+Y)/(1.+INV)
      IF(ABS(TEMP).GT.1 .OR. ABS(TEMP1).GT.1) THEN
              WRITE(6,*) 'FTT ERROR INV,Y=',INV,Y,TEMP,TEMP1
              FTT=1
              RETURN
      ENDIF
      FTTV=SQRT(.5+INV)*.5*(ASIN(TEMP) + ASIN(TEMP1))
      END
```

E. EXEC PROGRAM

```
CP LINK MVS 103 103 RR
ACC 103 K
*********************************
FIL * CLEAR
FIL 01 K DSN F0968 MCOR87 DATA (RECFM FB LRECL 69 BLOCK 17940
FIL 02 DISK  MC87 TEMP
FIL 06 &1 (RECFM FBA LRECL 150
FIL 25 DISK  MCLASS PG15    (RECFM F LRECL 25
FIL 27 DISK  MCLASS PG17    (RECFM F LRECL 25
FIL 29 DISK  MCLASS PG19    (RECFM F LRECL 25
FIL 30 DISK  MCLASS PG20    (RECFM F LRECL 25
FIL 31 DISK  MCLASS PG21    (RECFM F LRECL 25
FIL 32 DISK  MCLASS PG22    (RECFM F LRECL 25
&BEGSTACK
30.0        /* AVG INV THRESHOLD T */
30          /* NO. CELLS THRESHOLD K */
13          /* MOS (ONLY 1) */
4           /* YCS (ONLY 1) */
15          /* GRADE (ONLY 1) */
3 1 2 3     /* NO. SVC COMPS AND ARRAY(1-REG,2-AUGREG,3-RES,4=1+2,5=ALL */
1 16        /* NO. COMM SRCS AND ARRAY(1-15, 16=ALL)
1           /* 3RD DIMENSION (0=NONE, 1=SVC, 2=CS)
&END
LOAD MC87 (START CLEAR
```

F. SAMPLE DATA FILE

1	2	3	4	5	6	7
15	0	1	1	15	1	0.02
15	0	3	2	9	5	0.42
15	0	3	3	9	6	0.75
15	0	3	2	10	4	0.15
15	0	3	1	11	1	0.10
15	0	3	3	10	2	0.15
15	0	3	3	3	1	0.02
15	0	3	3	4	1	0.02
15	0	4	2	9	6	0.87
15	0	4	2	10	2	0.20
15	0	4	3	9	2	0.07
15	0	4	1	11	1	0.10
15	0	4	3	4	2	0.05
15	0	4	3	10	1	0.10
15	0	5	2	9	3	0.27
15	0	5	2	10	2	0.20
15	0	5	1	11	1	0.07
15	0	6	2	9	1	0.02
15	1	2	3	15	1	0.05
15	1	3	1	1	2	0.20
15	1	3	3	15	1	0.05
15	1	3	3	10	1	0.10
15	1	3	3	7	3	0.12
15	1	3	3	3	5	0.30
15	1	3	3	5	2	0.65
15	1	3	3	2	5	0.45
15	1	3	3	9	3	0.12
15	1	3	2	15	1	0.05
15	1	3	1	11	3	0.12
15	1	3	1	10	2	0.10
15	1	3	2	7	1	0.02
15	1	3	3	6	1	0.10
15	1	4	3	5	2	0.12
15	1	4	3	2	5	0.40
15	1	4	2	3	2	0.12
15	1	4	1	1	1	0.02
15	1	4	2	9	2	0.07
15	1	4	3	9	1	0.02
15	1	4	3	7	5	0.25
15	1	4	3	10	1	0.02
15	1	4	3	3	2	0.05
15	1	4	1	10	1	0.02
15	1	4	3	12	1	0.02
15	1	4	2	12	1	0.07

(remaining entries omitted)

Column descriptions:

1 - grade
2 - MOS
3 - YCS
4 - service component
5 - commissioning source
6 - number of records
7 - total average inventory

```
C --- PROGRAM TO CREATE INVENTORY DATA FILE BY GRADE
      PARAMETER (MXX=20000, MXY=10)
C --- CLASSIF. TABLE: GRADE, MOS, YCS, SVC, CS
      INTEGER*2 PTRTBL(MXX, 5), NRECS(MXX)
      REAL AINV(MXX)
      INTEGER TYPE,YCS,PG,MOS,SEX,CS,EDLV,SVC,MOS1,MOS2,RACE
      INTEGER DATA(MXY), SPG
      CHARACTER*7 CITLS
      DATA AINV/MXX*0./, NRECS/MXX*0/
C
      WRITE(5,*) 'ENTER PG'
      READ(5,*) SPG
      WRITE(5,*) 'PG TO USE=',SPG
      ICR=0
      NRC=0
      NG=0
      DO 10  I=1,999999
       READ(1,100,END=999) TYPE,YCS,PG,MOS,SEX,CS,EDLV,SVC,MOS1,MOS2,
     *        RACE,CITLS,DATA
       ICR=ICR+1
C ---  CLASSIFY ALL RECORDS TYPE 0
       IF(TYPE.GT.0) GO TO 999
C ---  ADD NEW RECORD TO TABLE
       IF(PG.EQ.SPG) CALL ADDTBL(PG,MOS,YCS,SVC,CS, DATA,MXY, PTRTBL,
     *                             MXX, NRC,AINV,NRECS)
       IF(PG.EQ.SPG) NG=NG+1
       IF(MOD(ICR,5000).EQ.0) WRITE(6,*) 'ICR,NRC=',ICR,NRC
   10 CONTINUE
C
  999 CONTINUE
      WRITE(6,*) ' '
      WRITE(6,*) 'TOTAL RECORDS READ           =',ICR
      WRITE(6,*) 'TOTAL RECORDS ACCEPTED       =',NG
      WRITE(6,*) 'TOTAL INVENTORY COMBINATIONS  =',NRC
      DO 20 I=1,NRC
       WRITE(2,101) (PTRTBL(I,J),J=1,5), NRECS(I),AINV(I)
   20 CONTINUE
  100 FORMAT(3I2,I3,I1,I2,2I1,2I3,I1,A7, 1X, 10I4)
  101 FORMAT(I2,I4,I3,I2,I3, I4, F7.2)
      END
C
      SUBROUTINE ADDTBL(PG,MOS,YCS,SVC,CS, DATA,MXY, PTRTBL,MXX, NRC,
     *        AINV,NRECS)
C --- SET INVENTORY POINTER FOR THIS ENTRY AND ACCUMULATE
      INTEGER*2 PTRTBL(MXX, 5), NRECS(MXX)
      REAL AINV(MXX)
      INTEGER YCS,PG,MOS,CS,SVC
      INTEGER DATA(MXY)
      MINV=GETINV(PTRTBL, MXX,NRC, PG,MOS,YCS,SVC,CS)
      IF(MINV .EQ. 0) THEN
C ---        NEW COMBINATION
            NRC=NRC+1
            IF(NRC .GT. MXX) THEN
              WRITE(6,*) '*** ERROR - TOO MANY INV. COMBINATIONS',NRC
              STOP
            ENDIF
```

```
            MINV=NRC
            PTRTBL(MINV, 1)=PG
            PTRTBL(MINV, 2)=MOS
            PTRTBL(MINV, 3)=YCS
            PTRTBL(MINV, 4)=SVC
            PTRTBL(MINV, 5)=CS
            NRECS(MINV)=0
      ENDIF
      AI=0
      DO 110 IT=1,MXY
       AI=AI + FLOAT(DATA(IT))
  110 CONTINUE
      AINV(MINV)=AINV(MINV) + .25*AI/MXY
      NRECS(MINV)=NRECS(MINV) + 1
      END
C ---
      FUNCTION GETINV(PTRTBL, MXX,NRC, PG,MOS,YCS,SVC,CS)
C --- FIND LOCATION OF MATCHING INVENTORY ENTRY FOR A LOSS
      INTEGER*2 PTRTBL(MXX, 5)
      INTEGER YCS,PG,MOS,CS,SVC
      DO 10 I=1,NRC
        IF(PTRTBL(I, 1) .EQ. PG  .AND.
     *     PTRTBL(I, 2) .EQ. MOS .AND.
     *     PTRTBL(I, 3) .EQ. YCS .AND.
     *     PTRTBL(I, 4) .EQ. SVC .AND.
     *     PTRTBL(I, 5) .EQ. CS ) THEN
                                    GETINV=I
                                    RETURN
        ENDIF
   10 CONTINUE
      GETINV=0
      END
```

APPENDIX C. SAMPLE OUTPUT

A. GENERAL

This appendix contains sample output from the computer program. A sample output for test cases one through 30 which use the first five estimation methods is shown in paragraph B. A sample output for the vector test cases is shown in paragraph C. These examples show the output that is produced by the WRITE statements for file definition 11, e.g., WRITE(11,101). The program also contains several WRITE and PRINT statements that provide interactive information to the user via the terminal screen, e.g., WRITE(6,*), WRITE(5,*) and PRINT *. This interactive output is omitted.

B. SAMPLE OUTPUT (TEST CASES 1-30)

```
TEST CASE INPUT PARAMETERS:
INVENTORY THRESHOLD= 30.0        THRESHOLD NO. OF CELLS= 30
MOS=  13      YCS=  4      GRADE= 15
SERVICE COMPONENTS=   1  2  3
COMM SOURCES=   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15

EXPANSION INFORMATION:
ACTUAL NO. OF CELLS USED= 24
MOS GROUP # 1 YCS'S USED=
  4  5
LARGE MOS GROUP #1 YCS'S USED=
  4  5
MAJOR MOS GROUP #1 YCS'S USED=
  4  5

EMP BAYES TRANS SCALE - TIME DEP VAR:

MEAN ABSOLUTE DEVIATION (ORIG SCALE):
                FRACTION CELLS  FRACTION  MAD
VALID YR    K   WITH  UNDERAGE  FROM UNDERAGE   MAD
--------  ----  --------------  -------------  -----
    1      24       0.458           0.478      0.127
    2      24       0.250           0.187      0.099
    3      24       0.542           0.441      0.098
    4      24       0.333           0.375      0.069
    5      24       0.417           0.352      0.072
    6      24       0.125           0.053      0.082
    7      24       0.208           0.138      0.099
    8      24       0.417           0.472      0.077
    9      24       0.833           0.943      0.181
   10      24       0.833           0.952      0.113
                                     AVG MAD = 0.102
```

CHI SQUARE (ORIG SCALE):
MIN CHI = 48.590 K = 24 VALID YR = 8
MAX CHI = 329.334 K = 24 VALID YR = 9
AVG CHI = 98.791

MEAN SQUARED ERROR (TRANS SCALE):
MIN MSE = 0.033 K = 24 VALID YR = 4
MAX MSE = 0.205 K = 24 VALID YR = 9
AVG MSE = 0.079

EMP BAYES TRANS SCALE - TIME INDEP VAR:

MEAN ABSOLUTE DEVIATION (ORIG SCALE):

VALID YR	K	FRACTION CELLS WITH UNDERAGE	FRACTION MAD FROM UNDERAGE	MAD
1	24	0.458	0.504	0.127
2	24	0.250	0.204	0.094
3	24	0.583	0.477	0.101
4	24	0.375	0.432	0.071
5	24	0.417	0.396	0.073
6	24	0.083	0.061	0.076
7	24	0.250	0.152	0.094
8	24	0.458	0.517	0.075
9	24	0.792	0.952	0.183
10	24	0.875	0.961	0.118
			AVG MAD =	0.101

CHI SQUARE (ORIG SCALE):
MIN CHI = 45.452 K = 24 VALID YR = 6
MAX CHI = 344.445 K = 24 VALID YR = 9
AVG CHI = 99.284

MEAN SQUARED ERROR (TRANS SCALE):
MIN MSE = 0.036 K = 24 VALID YR = 4
MAX MSE = 0.213 K = 24 VALID YR = 9
AVG MSE = 0.078

EMP BAYES ORIG SCALE - TIME DEP VAR:

MEAN ABSOLUTE DEVIATION (ORIG SCALE):

VALID YR	K	FRACTION CELLS WITH UNDERAGE	FRACTION MAD FROM UNDERAGE	MAD
1	24	0.458	0.478	0.127
2	24	0.250	0.195	0.097
3	24	0.542	0.461	0.093
4	24	0.333	0.397	0.070
5	24	0.417	0.375	0.074
6	24	0.125	0.057	0.079
7	24	0.208	0.161	0.102
8	24	0.417	0.488	0.079
9	24	0.833	0.944	0.185
10	24	0.833	0.952	0.117
			AVG MAD =	0.102

CHI SQUARE (ORIG SCALE):
MIN CHI = 49.207 K = 24 VALID YR = 6
MAX CHI = 340.035 K = 24 VALID YR = 9
AVG CHI = 100.985

EMP BAYES ORIG SCALE - TIME INDEP VAR:

MEAN ABSOLUTE DEVIATION (ORIG SCALE):

VALID YR	K	FRACTION CELLS WITH UNDERAGE	FRACTION MAD FROM UNDERAGE	MAD
1	24	0.458	0.479	0.127
2	24	0.250	0.196	0.097
3	24	0.542	0.462	0.093
4	24	0.333	0.399	0.070
5	24	0.417	0.377	0.075
6	24	0.125	0.056	0.079
7	24	0.208	0.161	0.101
8	24	0.417	0.490	0.079
9	24	0.833	0.945	0.185
10	24	0.833	0.953	0.117
			AVG MAD =	0.102

CHI SQUARE (ORIG SCALE):
MIN CHI = 48.836 K = 24 VALID YR = 6
MAX CHI = 339.835 K = 24 VALID YR = 9
AVG CHI = 100.960

EFRON-MORRIS TRANS SCALE - TIME DEP VAR:

MEAN ABSOLUTE DEVIATION (ORIG SCALE):

VALID YR	K	FRACTION CELLS WITH UNDERAGE	FRACTION MAD FROM UNDERAGE	MAD
1	24	0.458	0.475	0.126
2	24	0.250	0.172	0.095
3	24	0.583	0.451	0.102
4	24	0.375	0.365	0.076
5	24	0.333	0.348	0.076
6	24	0.083	0.046	0.082
7	24	0.208	0.135	0.098
8	24	0.458	0.469	0.076
9	24	0.708	0.938	0.185
10	24	0.875	0.960	0.115
			AVG MAD =	0.103

CHI SQUARE (ORIG SCALE):
MIN CHI = 46.301 K = 24 VALID YR = 8
MAX CHI = 340.712 K = 24 VALID YR = 9
AVG CHI = 101.231

MEAN SQUARED ERROR (TRANS SCALE):
MIN MSE = 0.042 K = 24 VALID YR = 4
MAX MSE = 0.211 K = 24 VALID YR = 9
AVG MSE = 0.080

C. SAMPLE OUTPUT (VECTOR TEST CASES)

```
TEST CASE INPUT PARAMETERS:
INVENTORY THRESHOLD= 30.0       THRESHOLD NO. OF CELLS= 30
MOS= 151     YCS=  7     GRADE= 17
SERVICE COMPONENTS=   1  2  3
COMM SOURCES=   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15

EXPANSION INFORMATION:
ACTUAL NO. OF CELLS USED=  8
MOS GROUP # 8 YCS'S USED=
  7
LARGE MOS GROUP #3 YCS'S USED=
  7
MAJOR MOS GROUP #2 YCS'S USED=
  7


EMP BAYES TRANS SCALE - VECTOR CASE:

VECTOR IS BY SERVICE COMPONENT
K=  8     P=  3     KP=  24

MEAN ABSOLUTE DEVIATION (ORIG SCALE):
                FRACTION CELLS  FRACTION  MAD
VALID YR   KP   WITH  UNDERAGE  FROM UNDERAGE   MAD
--------   ----  --------------  -------------  -----
    1       24        0.375          0.219       0.186
    2       24        0.458          0.375       0.179
    3       24        0.458          0.417       0.130
    4       24        0.458          0.422       0.127
    5       24        0.542          0.708       0.146
    6       24        0.292          0.310       0.142
    7       24        0.250          0.193       0.126
    8       24        0.375          0.434       0.092
    9       24        0.458          0.705       0.161
   10       24        0.792          0.912       0.202

CHI SQUARE (ORIG SCALE):
MIN CHI =    27.827     KP =  24     VALID YR =  8
MAX CHI =   165.694     KP =  24     VALID YR = 10
AVG CHI =    61.025

MEAN SQUARED ERROR (TRANS SCALE):
MIN MSE =  0.089     KP =  24     VALID YR =  8
MAX MSE =  0.483     KP =  24     VALID YR = 10
AVG MSE =  0.229
```

REFERENCES

Amin Elseramegy, H., *CART Program: Implementation of the CART Program and Its Application to Estimating Attrition Rates*, Master's Thesis, Naval Postgraduate School, Monterey, California, December 1985.

Carter, G.M. and Rolph, J.E., "Empirical Bayes Methods Applied to Estimating Fire Alarm Probabilities," *Journal of the American Statistical Association*, v.69, pp. 880-885, December 1974.

Casella, G., "An Introduction to Empirical Bayes Data Analysis," *The American Statistician*, v.39, pp. 83-87, May 1985.

Decision System Associates, Inc., *Functional Description for the Development of the Officer Planning and Utilization System (OPUS)*, Rockville, Maryland, 1986.

Dickinson, C.R., *Refinement and Extension of Shrinkage Techniques in Loss Rate Estimation of Marine Corps Officer Manpower Models*, Master's Thesis, Naval Postgraduate School, Monterey, California, March 1988.

Efron, B. and Morris, C., "Empirical Bayes on Vector Observations: An Extension of Stein's Method," *Biometrika*, v.59, pp. 335-347, 1972.

Efron, B. and Morris, C., "Stein's Estimation Rule and Its Competitors - An Empirical Bayes Approach," *Journal of the American Statistical Association*, v.68, pp. 117-130, March 1973.

Efron, B. and Morris, C., "Data Analysis Using Stein's Estimator and Its Generalizations," *Journal of the American Statistical Association*, v.70, pp. 311-319, June 1975.

Hogan, D.L. Jr., *The Use of Exponential Smoothing to Produce Yearly Updates of Loss Rates Estimates in Marine Corps Manpower Models*, Master's Thesis, Naval Postgraduate School, Monterey, California, June 1986.

Larsen, R.W., *The Aggregation of Population Groups to Improve the Predictability of Marine Corps Officer Attrition Estimation*, Master's Thesis, Naval Postgraduate School, Monterey, California, December 1987.

Marine Corps Order P1200.7G, *Military Occupational Specialties Manual (MOS Manual)*, Chapter 1, 1 April 1988.

Naval Postgraduate School Report NPS55-88-006, *The Use of Shrinkage Techniques in the Estimation of Attrition Rates for Large Scale Manpower Models*, R.R. Read, 27 July 1988.

Navy Personnel Research and Development Center, *System Design for the Marine Corps Officer Rate Projector (MCORP)*, San Diego, California, 1985.

Robinson, J.R., *Limited Translation Shrinkage Estimation of Loss Rates in Marine Corps Manpower Models*. Master's Thesis, Naval Postgraduate School, Monterey, California, March 1986.

Siegel, B., *Methods for Forecasting Officer Loss Rates*, Navy Personnel Research and Development Center, San Diego, California, 1983.

Tucker, D.D., *Loss Rate Estimation in Marine Corps Officer Manpower Models*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1985.

Yacin, N., *Application of Logistic Regression to the Estimation of Manpower Attrition Rates*, Master's Thesis, Naval Postgraduate School, Monterey, California, March 1987.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Professor Robert R. Read, (Code 55Re) Naval Postgraduate School Monterey, CA 93943-5000	5
4.	Professor Gerald G. Brown, (Code 55Bw) Naval Postgraduate School Monterey, CA 93943-5000	1
5.	Major John M. Misiewicz MAGTF Warfighting Center (WF13) Marine Corps Combat Development Command Quantico, VA 22134-5001	5
6.	Commanding Officer Attn: Carol Mullins Navy Personnel Research and Development Center San Diego, CA 92152	1
7.	Mr. R. Morton Decision System Associates, Inc. 350 Fortune Terrace Rockville, MD 20854-2995	1
8.	Commandant of the Marine Corps Code MI Attn: Major D. O'Dell Headquarters, U.S. Marine Corps Washington, D.C. 20380-0001	1
9.	Commandant of the Marine Corps Code MPP-31 Attn: Major R.W. Larsen Headquarters, U.S. Marine Corps Washington, D.C. 20380-0001	1

10. Director 1
Center for Naval Analysis
4401 Ford Avenue
Alexandria, VA 22302

11. Commandant of the Marine Corps 1
Code TE06
Headquarters, U.S. Marine Corps
Washington, D.C. 20380-0001

12. Captain Charles R. Dickinson 1
Staff USCINCPAC Box 15(J55)
Camp H.M. Smith, HI 96861

13. Captain Michael J. Streff 1
HQDA
Attn: DAPE-MBB-P (Room 2D669)
Washington, D.C. 20310

Thesis
M63653 Misiewicz
c.1 Extension of aggregation and shrinkage techniques used in the estimation of Marine Corps Officer attrition rates.